File Number S360-28
Form C24-3387-3

BOS

IBM Systems Reference Library

IBM System/360 Basic Operating System Report Program Generator Specifications

This reference publication contains fundamentals of RPG programming and language specifications for the IBM System/360 Basic Operating System Report Program Generator. For information on the Basic Operating System that is beyond the purpose of this language publication, see IBM System/360 Basic Operating System Programmer's Guide, Form C24-3372.

For the titles and abstracts of associated publications, see the IBM System/360 Bibliography, Form A22-6822.

PREFACE

The System/360 Basic Operating System Report Program Generator (RPG) is a problem-oriented language designed to provide users with an efficient, easy-to-use technique for generating programs that can:

1. Obtain data records from single- or multiple-input files,
2. Perform calculations on data taken from input records or RPG literals,
3. Write printed reports,
4. Use Table Lookup,
5. Exit to a user's subroutine written in a language other than RPG,
6. Branch within the calculations, and
7. Sequence-check input records.

RPG uses a set of specifications sheets on which the user makes entries. The forms are simple, and the headings on the sheets are largely self-explanatory.

Although many reports use only one input file, RPG can combine data from multiple input files to create a report. The output may be a single report, or it may be several reports created simultaneously on different devices.

SIMPLIFIED TERMINOLOGY

The titles of some IBM programming systems have been simplified as shown here:

former: IBM System/360 Basic Operating System....(8K Disk)
new: IBM System/360 Basic Operating System

former: IBM System/360 Basic Operating System....(16K Tape)
new: IBM System/360 Tape Operating System

former: IBM System/360 Basic Operating System....(16K Disk)
new: IBM System/360 Disk Operating System

Major Revision, May 1966

This edition, C24-3387-3, is a major revision of and obsoletes Form C24-3387-2, and Technical Newsletters N24-5023 and N24-5037. Changes are designated in three ways:

1. A vertical line appears at the left of affected text where only a part of the page has been changed.
2. A dot (●) appears at the left of the page number where the complete page should be reviewed.
3. A dot (●) appears at the left of the title of each figure that has been chnaged.

Copies of this and other IBM publications can be obtained through IBM Branch Offices. A form has been provided at the back of this publication for readers' comments. If the form has been detached, comments may be directed to: IBM Programming Publications, Endicott, New York 13760.

© by International Business Machines Corporation 1965

CONTENTS

RPG operates under control of the IBM System/360 Basic Operating System. The Basic Operating System provides the RPG compiler with input and output services. Object programs generated by the RPG compiler also operate under operating-system control and depend on it for similar services.

RPG supports the minimum configuration required by the Basic Operating System. The Decimal Arithmetic feature is required for RPG.

COMPATIBILITY OF BASIC OPERATING SYSTEM RPG WITH MODEL 20 RPG OR WITH SYSTEM/360 BASIC PROGRAMMING SUPPORT RPG

The RPG source language is upward-compatible. The Basic Operating System Report Program Generator can compile a Model 20 RPG source program, or a Basic Programming Support RPG source program that adheres to its language specifications and compatibility statements. The object program will produce equivalent results, if the same input/output facilities exist.

- The File Description Specification sheet of the Basic Operating System RPG may require additional information (record length and overflow indicators).
- The control card of the Model 20 RPG or Basic Programming Support RPG programs must be modified before the object program can be generated by the Basic Operating System RPG.

FUNCTION OF RPG

When RPG is used, the IBM System/360 actually performs two separate functions:

1. Program-generating, and
2. Data processing.

In program-generating, program specifications are defined by the user to produce machine-language instructions. Storage areas are automatically assigned, constants or other reference factors are included, and linkages to routines are produced for checking, for input/output operations, and for other functions.

In data processing, the machine-language instructions (created in the first function) are combined with the user input-data files, and both are processed through the system to produce the desired reports or output files.

USING RPG

The preparation of a report by RPG consists of the general operations illustrated in Figure 1 and described as follows:

1. The programmer must evaluate the report requirements to determine the format of the input files and the appearance of the finished report. For example, he must know what fields in the input files are to be used, what kind of calculations are to take place, the location of the data in the output records, and the number and the kind of totals that must be accumulated. More specific information regarding the evaluation of report requirements is described in Problem Definition.

2. After the programmer has evaluated the requirements of the report, he provides the same information to the RPG program.

   He must describe his input (record layout, fields used, etc.) by making entries on an Input Specification sheet.

   He must state what processing is to be done (add, subtract, multiply) by entries on a Calculation Specification sheet.

   He must state how the finished report is to look (printing positions, carriage control, etc.) by making entries on the Output-Format Specification sheet.

   He must describe all files used by the object program (input files, output files, table files, etc.) by making entries on the File Description Specification sheet.

3. After the specifications have been written on the appropriate forms, cards are keypunched with the data from the forms.

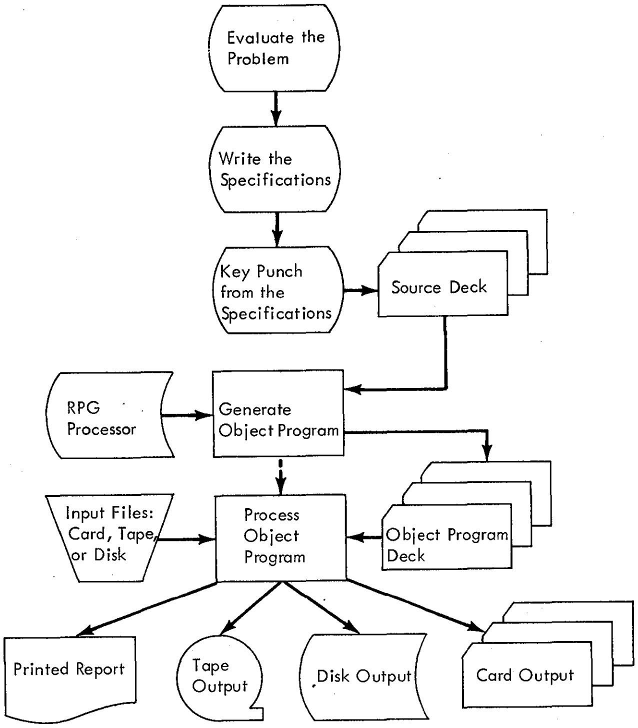


Figure 1. Producing Reports Using the RPG Program

4. These punched cards (called a source deck) are combined with the RPG Processor control card and the Basic Operating System Job Control statements. The source deck and the control cards are supplied to an input device and are processed under control of the Basic Operating System. At the end of this processing run (known as a generating or compiling run), a program capable of preparing the report specified by the programmer has been produced. This program (known as an object program) contains all of the computer instructions and linkages to the Control System necessary to prepare the desired report.

5. The input files can then be read into the system, and processing of the program will begin. This is known as the object run.

At the end of the object run, the report has been prepared and any other functions, such as file updating, are completed. Through facilities provided by the Basic Operating System, the object program can be retained for later runs without recompilation.

MACHINE FEATURES REQUIRED

PROGRAM GENERATION

1. 8K bytes of main storage (the Supervisor must not exceed 4K)

2. One of the following:

   IBM 1442 Card Read-Punch

   IBM 2501 Card Reader

   IBM 2520 Card Read-Punch

   IBM 2540 Card Read-Punch

3. One of the following:

   IBM 1403 Printer

   IBM 1404 Printer (continuous forms operation only)

   IBM 1443 Printer

4. IBM 2311 Disk Storage Drive

5. Standard Instruction Set

6. Decimal Arithmetic Feature

OBJECT PROGRAM EXECUTION

1. 8K bytes of main storage

2. I/O units as requested by the specifications

3. Standard Instruction Set

4. Decimal Arithmetic Feature

5. IBM 2311 Disk Storage Drive

ADDITIONAL MACHINE FEATURES SUPPORTED

PROGRAM GENERATION

1. 16K, 32K, or 64K bytes of main storage.

2. One IBM 2400-series Magnetic Tape Unit (7- or 9-track) (for source program input only).

OBJECT PROGRAM EXECUTION

1. 16K, 32K, or 64K bytes of main storage.

2. IBM 2311 Disk Storage Drive

3. IBM 2400-series Magnetic Tape Unit (7- or 9-track). If a 7-track tape unit is used, the Data Conversion feature is required.

Note: The object program produced by 8K BOS RPG does not have the capability of running with a non-resident supervisor.

For usage of tape units at either compile time or object time, refer to Appendix H of the programmer's guide publication listed on the front cover of this publication.

This section introduces some of the basic RPG functions. These functions are considered basic because they are probably used in the most typical reports produced by RPG, and because they depend on a single sequential input file only. They are also related to the functions commonly performed with conventional punched-card equipment.

Although RPG programs can process files contained on magnetic tape or direct-access storage devices, only card-input files are used for this introduction to RPG. Programmers familiar with the basic concepts of RPG may omit this section and concentrate on the detailed specifications descriptions contained in the section RPG Specification Sheets of this publication.

Because of the numerous fields on the five specification sheets, it may appear that writing RPG specifications is a difficult task. However, few programs use all the specifications, and some may require entries on only one or two lines of the sheets.

At the end of the discussion of basic RPG functions, a simple file-to-file listing application is used to illustrate how the specification sheets are related.

FUNCTIONS DESCRIBED

Fourteen of the most basic RPG functions are described in this section:

1. Describing a record and its fields to the system.
2. Adding and subtracting (crossfooting).
3. Detail printing.
4. Establishing control fields.
5. Total calculations.
6. Detail and total printing.
7. Group printing.
8. Group indication.
9. Overflow printing.
10. Testing for zero, plus and minus balance.
11. Using field indicators.
12. Comparing.
13. Multiplying and dividing.
14. Sequence-checking.

DESCRIBING A RECORD AND ITS FIELDS TO THE SYSTEM

Entries on the Input Specification sheet describe the data records and the fields to be read into the system.

Problem

A detail labor file contains card records as shown in Figure 2. Three fields (A, B, and C) are to be read from each card into main storage. Field A is contained in columns 46-50; Field B is contained in columns 56-60; and Field C is contained in columns 66-70 of each record. The file that contains the records and the fields within each record are described on the Input Specification sheet.

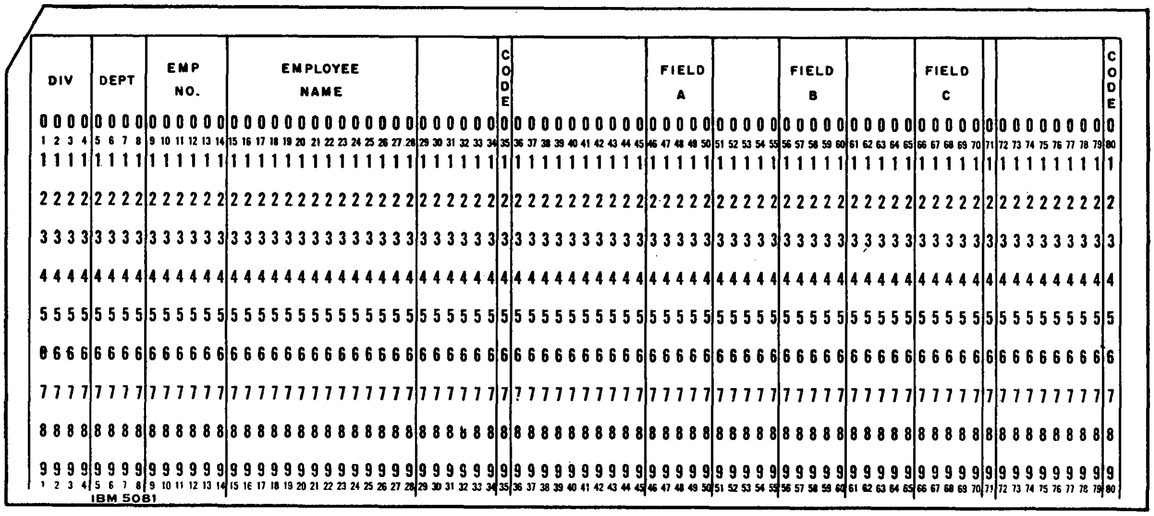


Figure 2. Input Cards for Detail Labor File

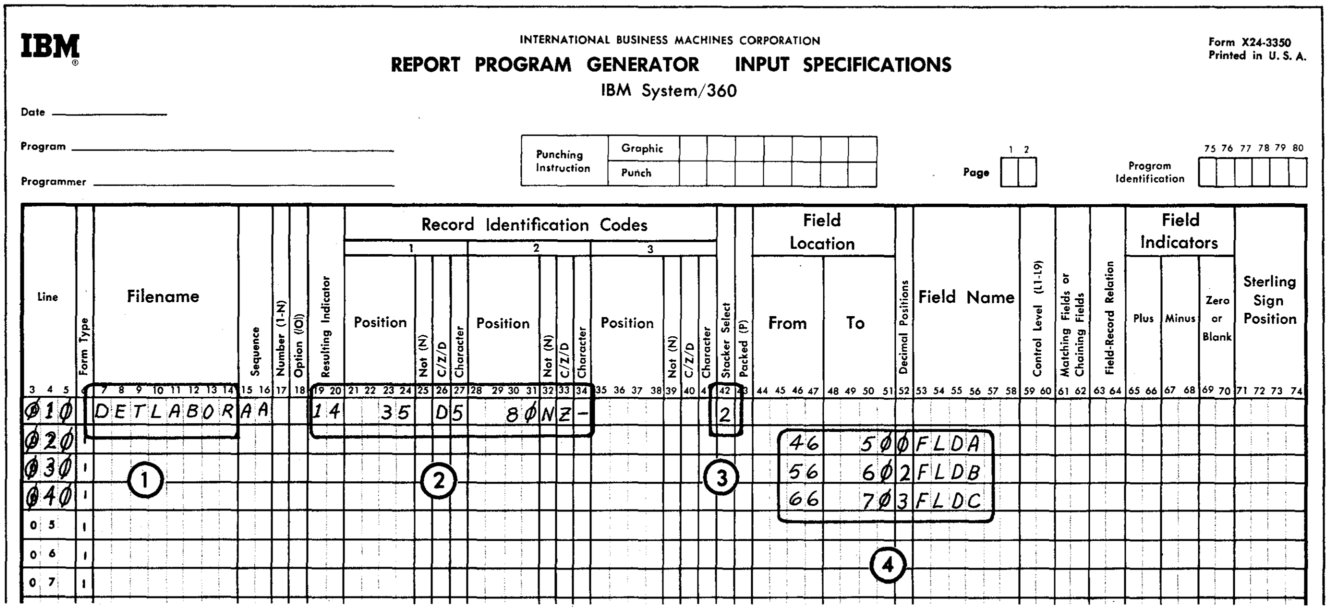
Each card that contains fields A, B, and C is identified by two distinct attributes. They must contain a digit 5 in column 35 and no 11-zone in column 80. When these two conditions exist in the card record, the record is valid (that is, it contains fields A, B, and C).

SPECIFICATIONS

The entries on the Input Specification sheet (Figure 3) that apply to this problem are circled and identified by numbers to reference the fields in text. The remaining fields on the form are described later.

1. Each file of cards must be given a name. In this example the input deck of cards represents a detail labor file (Figure 2). A file name can be used to describe only one file.

2. If several types of cards exist in a file, each card must be identified by its particular card code. In this example, each card read by the system must contain a 5-punch in column 35 and no 11-zone (minus) punch in column 80. These identifying codes are placed in



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS
IBM System/360

Form X24-3350
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1 Position	Not (N)	C/Z/D	Character	2 Position	Not (N)	C/Z/D	Character	3 Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	Field Location From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators Plus	Minus	Zero or Blank	Sterling Sign Position
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
Ø1Ø		DETLABOR	AA			14	35		D	5	8Ø	N	Z	-					2												
Ø2Ø																					46	5Ø	Ø	FLDA							
Ø3Ø	I																				56	6Ø	2	FLDB							
Ø4Ø	I																				66	7Ø	3	FLDC							
0 5	I																														
0 6	I																														
0 7	I																														

Figure 3. Input Specification Sheet

the fields called Record Identification Codes (columns 21-41 of the form). These fields provide for three identifying codes; however, more codes can be indicated (see Record Identification Codes).

In this example the card codes are specified by writing a 5 in Character and a 35 in Position for the first code; and by writing an 80 in Position and a minus in Character, and an N in Not for the second code. The field on the form called C/Z/D indicates how the card column containing the card code is to be compared: D = digit portion only, Z = zone portion only, or C = all portions of the card column.

After all the identification codes are established, the programmer assigns a two-digit number (from 01 to 99) to the card type identified. This code, known as a resulting indicator (columns 19-20 of the form), is used to refer to this specific card type on other specification sheets. Using this code reduces the number of entries required on other specification forms, as will be seen later.

If the input card is to be selected into a stacker (other than the one into which it would normally be selected), the stacker number is written in Stacker Select (Column 42).

Each field of the card to be read must be defined as shown on the lines following the record-identification line. The specifications shown in this example locate the data from the input record and place it in fields A, B and C (FLDA, FLDB, and FLDC) in three separate locations in main storage.

Once the card columns of the field have been specified and the appropriate field name has been defined for the field, other references to this field are made by using its field name, rather than writing down the specific card columns each time.

Although the input illustrated in Figure 3 is simplified, the entries shown will cause the contents of fields A, B, and C from the card to be read into the system during the processing of the object program. Calculations to be made from the input data are written on the Calculation Specification sheet.

ADDITION AND SUBTRACTION

Problem

In this problem, fields A, B, and C from the previous example are used to calculate a new field, Field D. The calculation A + B - C = D is to be performed. This operation is sometimes referred to as crossfooting. The form required for this operation is the Calculation Specification sheet.

Specifications

Figure 4 shows the required entries using the fields from the previous example. The first line of the Calculation Specification sheet tells the object program to add the contents of FLDB to the contents of FLDA and place the result in FLDD.

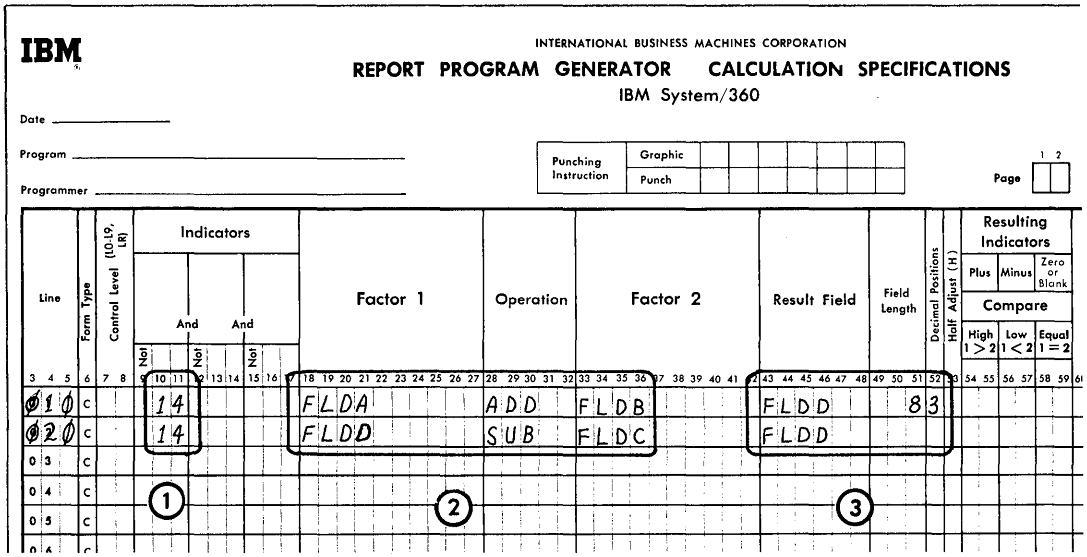
1. The 14 in Indicators defines the card type for which the calculations are to be performed, as assigned on the Input Specification sheet in Resulting Indicator (Figure 3).

2. The card columns for FLDA and FLDB were defined by entries in columns 46-47 and 50-51 on the input forms and need not be specified again. The names assigned in columns 53-56 will be used to refer to the fields.

3. The result field FLDD is defined for the program by merely writing the name FLDD in Result Field (columns 43-48) and indicating in Field Length (columns 49-51) the number of positions that must be set aside for this field in the object program. Just as in the case of FLDA and FLDB, which were defined on the input form, the name FLDD can now be used in other calculation operations or used in defining output specifications.

   Note: The result of A + B is not placed back into FLDA. To do this would destroy the original contents of FLDA. In this example it is necessary to save the original value of FLDA so that it can be printed on an output report.

   The second line in Figure 4 causes the object program to subtract the contents of FLDC from the result just previously attained in FLDD, and to store the new result back into FLDD.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Line	Form Type	Control Level (L0-L9, LR)	Indicators (Not)	And (Not)	And (Not)	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators Plus / Compare High 1>2	Minus / Low 1<2	Zero or Blank / Equal 1=2
3 4 5	6	7 8	9 10 11	12 13 14	15 16 17	18–27	28–32	33–42	43–48	49–51	52	53	54 55	56 57	58 59
∅1∅	C		14			FLDA	ADD	FLDB	FLDD	8	3				
∅2∅	C		14			FLDD	SUB	FLDC	FLDD						
0 3	C														
0 4	C														
0 5	C														
0 6	C														

Figure 4. Calculation Specification Sheet

DECIMAL POINT LOCATION: In operations involving numbers containing decimal positions, the number of decimal positions in each of the fields is frequently not the same. When the number of decimal positions is not the same in both factors in an arithmetic operation, shifting the factors to align the decimal point is usually required.

RPG automatically shifts the factors. The programmer must indicate the number of decimal positions contained in each factor used in calculations, and the number of decimal positions required in the result. If a field is to have arithmetic operations performed on it, the decimal positions must be specified. A zero is coded for no decimal positions (see Figure 4). Assume that the fields in Figure 3 have decimal positions as indicated under Decimal Position (column 52) in the following example:

Field From	Field To	Decimal Positions	Field Name
46	50	0	FLDA
56	60	2	FLDB
66	70	3	FLDC

A set of values for these fields might appear to the program as follows:

Field	Decimal Positions	Contained In Cards	Actual Value
A	0	00126	126.
B	2	01123	11.23
C	3	04264	4.264

The programmer must indicate the number of decimal positions required in the result.

The calculation A + B - C = D would result in the values being shifted like this:

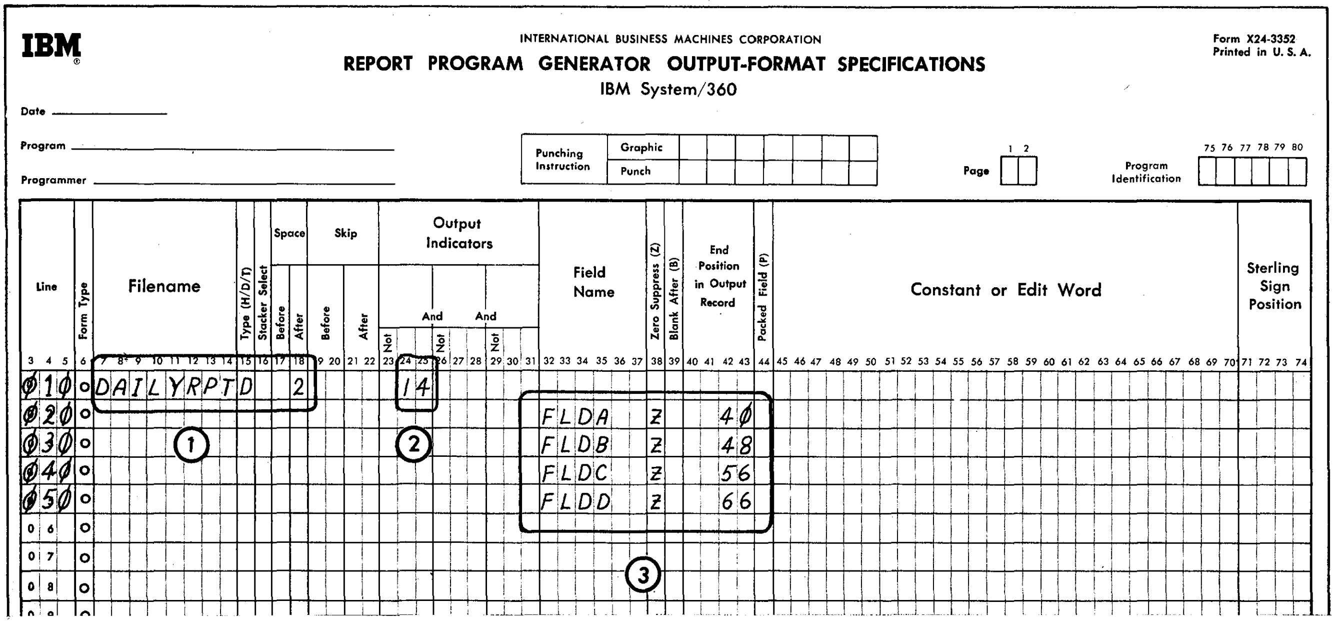
```
126.000 = A
+11.230 = B
137.230
- 4.264 = C
132.966 = D
```

(The result field has three decimal positions).

The result 132.966 is stored as the value of the field called D. Entering the decimal positions on the input sheet and the calculation sheet enables the factors to be shifted automatically.

DETAIL PRINTING

Detail printing is the printing of information obtained from each record as it is read.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS
IBM System/360

Form X24-3352
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
Ø1Ø	O	DAILYRPT	D			2			14							
Ø2Ø	O									FLDA	Z		4Ø			
Ø3Ø	O									FLDB	Z		48			
Ø4Ø	O									FLDC	Z		56			
Ø5Ø	O									FLDD	Z		66			
06	O															
07	O															
08	O															

Figure 5. Output-Format Specification Sheet

Problem

This problem illustrates how input fields FLDA, FLDB, and FLDC, and the calculated result, FLDD, can be specified for listing.

Specifications

Figure 5 shows the output specifications required. The numbers in the following list refer to items circled in Figure 5.

1. The output listing must be assigned its own specific file name. In this example it has been called DAILYRPT.

   The D in Type H/D/T indicates that the line being printed is a detail line. That is, it contains information from the record just read. (The other possible entries, H and T, indicating heading lines and total lines, are described later.)

   The 2 in Space After provides a double-spaced listing. (There is one blank line after each printed line.)

2. The 14 in Output Indicators identifies the input record type to be printed.

3. Each field to be printed must be specified under the heading Field Name. The Z in Zero Suppress means that zeros to the left of significant digits are not printed. (For example, the value stored in FLDA (00126) is printed as 126.)

   The printing positions for data to be printed on the output report are specified in End Position in Output Record (Columns 40-43). The programmer has to indicate only the last printing position of each field (low-order or rightmost position). The length of each field has already been established for the program as part of the input specifications, or established in the result field of the calculations specifications.

When preparing a simple listing such as the one in this example, the programmer would probably select printing positions of the output unit based upon the number of positions in each field plus a number of positions as spaces between fields.

More elaborate reports are laid out on a printer form in advance of the program writing function. Usually this indicates both the location on the form where data is to be printed and the source of the data (the card columns of the input data or the names of data fields developed in the program). This is normally a function of job definition, a subject that is discussed later under Problem Definition.

Figures 3, 4, and 5 illustrate how to read data into the system, how calculations can be performed upon the data, and how the original data and the data developed in the program can be printed. The examples so far have shown the items necessary to prepare a report. A significant item not yet described is how to specify a control field.

CONTROL FIELDS

A control field is a field containing information to be compared from record to record. A control break occurs when the information in the control field changes. A control level establishes the relative importance of the control fields.

Problem

This example shows how to specify three levels of control.

Specifications

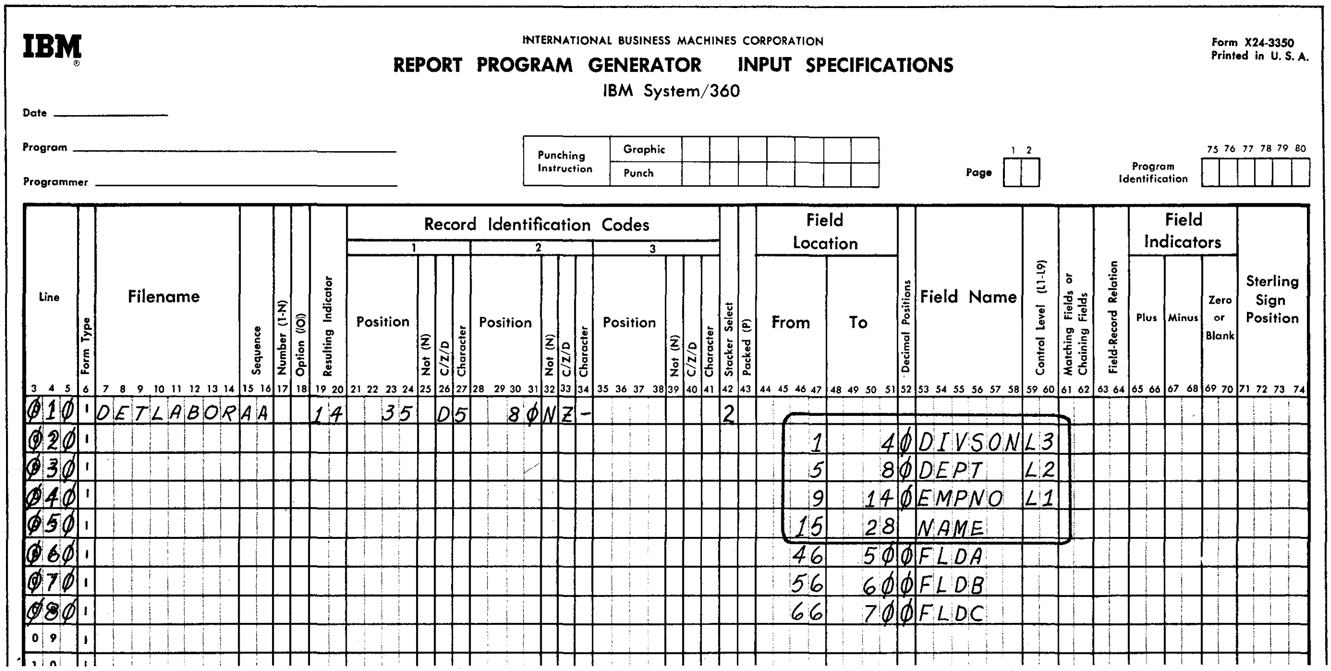
Figure 6 shows the input data from Figure 3 with the addition of the appropriate control data and employee name (the items inside the circle). In this example there are three levels of totals: the lowest level is employee number (EMPNO), the next is department (DEPT), and the highest level is division (DIVSON). These levels are designated L1, L2, and L3, respectively. (Because as many as nine levels of totals are possible in RPG, the common terms of minor, intermediate, and major totals are inadequate.)

To designate a control field and to establish a level of control, enter the card columns of the field in Field Location, enter an appropriate name in Field Name, and enter the correct control level (L1, L2, . . . L9) in Control Level.

In the example (Figure 2 and 6) the sequence of the control fields (left to right) in the card is the same as the sequence of the control levels. The control fields, however, could have been in any location on the card, and the specifications for them could have been in any sequence on the form, as illustrated in the following example.

The reading of the employee-name field (fifth line of the sheet) is also included as part of this example, but it is not related to controls. It is shown here to indicate another aspect of the decimal-positions column of the Input Specification sheet. When a numeric field is specified in Field Location, any digit (0 to 9) placed in Decimal Position causes zone punches in all positions of the field (except the units position) to be removed.

Therefore, when an alphabetic field is specified, Decimal Position should be left blank to permit the zone punches of the alphabetic field to be read into the system.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS
IBM System/360

Form X24-3350
Printed in U. S. A.

Date

Program

Programmer

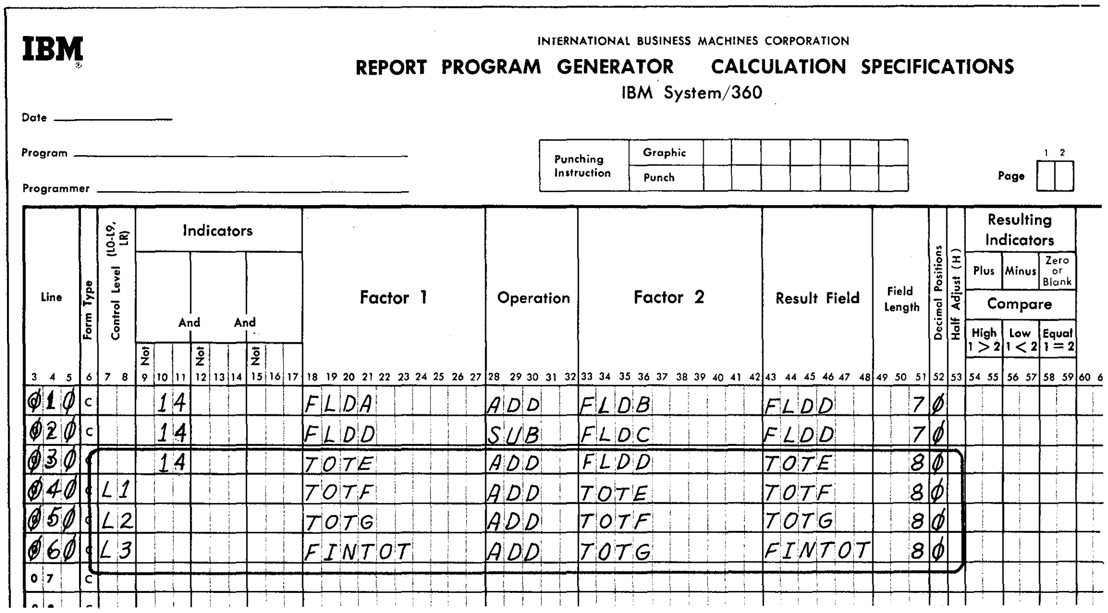
Punching Instruction	Graphic							
	Punch							

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1 Position	Not (N)	C/Z/D	Character	2 Position	Not (N)	C/Z/D	Character	3 Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators Plus	Minus	Zero or Blank	Sterling Sign Position
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
Ø1Ø	I	DETLABOR	AA			14	35		D	5	8Ø	N	Z	-					2												
Ø2Ø	I																				1	4	Ø	DIVSON	L3						
Ø3Ø	I																				5	8	Ø	DEPT	L2						
Ø4Ø	I																				9	14	Ø	EMPNO	L1						
Ø5Ø	I																				15	28		NAME							
Ø6Ø	I																				46	5Ø	Ø	FLDA							
Ø7Ø	I																				56	6Ø	Ø	FLDB							
Ø8Ø	I																				66	7Ø	Ø	FLDC							
0 9	I																														

Figure 6. Specifying Control Levels on the Input Sheet



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Line	Form Type	Control Level (L0-L9, LR)	Indicators		And		And		Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Compare: High 1 > 2	Minus / Low 1 < 2	Zero or Blank / Equal 1 = 2
3 4 5	6	7 8	Not 9	10 11	Not 12	13 14	Not 15	16 17	18–27	28–32	33–42	43–48	49 50 51	52	53	54 55	56 57	58 59
Ø1Ø	C			14					FLDA	ADD	FLDB	FLDD	7	Ø				
Ø2Ø	C			14					FLDD	SUB	FLDC	FLDD	7	Ø				
Ø3Ø	C			14					TOTE	ADD	FLDD	TOTE	8	Ø				
Ø4Ø	C	L1							TOTF	ADD	TOTE	TOTF	8	Ø				
Ø5Ø	C	L2							TOTG	ADD	TOTF	TOTG	8	Ø				
Ø6Ø	C	L3							FINTOT	ADD	TOTG	FINTOT	8	Ø				
0 7	C																	

Figure 7. Specifying Control Levels on the Calculation Sheet

TOTAL CALCULATIONS

Total calculations are performed after a specified control break has occurred. When a control break occurs, special operations are normally performed before processing the record that caused the control break. These operations are called <u>total operations</u>.

Problem

This example illustrates how totals can be accumulated at each control level.

Specifications

Figure 7 contains the data shown in Figure 4. The additional information for this example is circled. During processing of the object program, lines 1 and 2 cause the operation A + B - C = D. The third line adds the result, contained in FLDD, from each detail card to a field called TOTE. Therefore, TOTE is the accumulated amount for each employee group. The first three specification lines are performed in the object program each time a detail card is read (see Figure 7, Indicator 14).

When the level-1 (L1) control break occurs (a card from the next employee group has been sensed), specification-line 4 adds the total of TOTE accumulated from each detail card into a field called TOTF. TOTF is used to accumulate the total employee group for each department.

When the level-2 control break occurs, TOTF is added into TOTG.

When the level-3 control break occurs, TOTG (which represents the accumulated amounts for each division) is added into FINTOT.

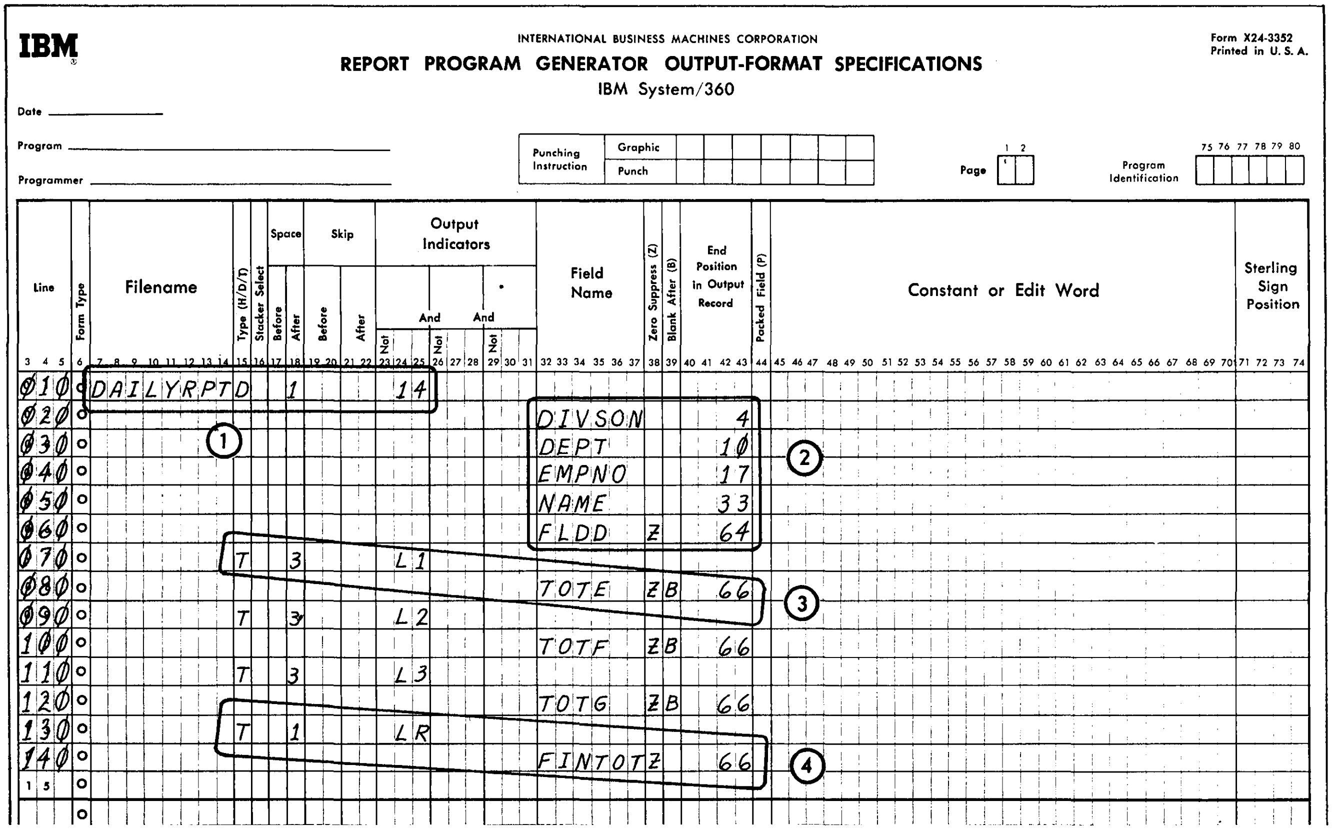
Accumulating totals at each control-break level is normally done when the corresponding totals are printed on an output report.

> Note: In System/360 RPG (as in IBM punched-card data processing equipment) a control break at one level forces control breaks for all lower levels.

DETAIL AND TOTAL PRINTING

Problem

This example shows the specifications necessary to print the three controlling fields, the name, the accumulated amount in field D, the three accumulated totals, and the final total at the end of the object-program report.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page

Program Identification

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
Ø1Ø	O	DAILYRPT	D			1			14							
Ø2Ø	O									DIVSON			4			
Ø3Ø	O									DEPT			1Ø			
Ø4Ø	O									EMPNO			17			
Ø5Ø	O									NAME			33			
Ø6Ø	O									FLDD	Z		64			
Ø7Ø	O		T			3			L1							
Ø8Ø	O									TOTE	Z	B	66			
Ø9Ø	O		T			3			L2							
1ØØ	O									TOTF	Z	B	66			
11Ø	O		T			3			L3							
12Ø	O									TOTG	Z	B	66			
13Ø	O		T			1			LR							
14Ø	O									FINTOT	Z		66			

Figure 8. Specifications on the Output-Format Sheet

Specifications

Figure 8 illustrates the specifications required for the report shown in Figure 9. The following numbers refer to the items circled in Figure 8:

1. This information is similar to the specifications in the previous example in Figure 5.

2. The data fields DIVSON, DEPT, EMPNO, NAME, and FLDD, are specified for printing.

3. The specifications to print the total for the first control level shown on lines 7 and 8 are:

   a. The T in H/D/T (column 15) indicates that the line is a total line. A total line is an operation caused by a control break. The input record that causes the control break cannot contribute data to the accumulated totals or to the total line. Totals accumulated before a control break are printed.

   b. A 3 in Space After (column 18) provides two blank lines after each printed line to make the report easier to read.

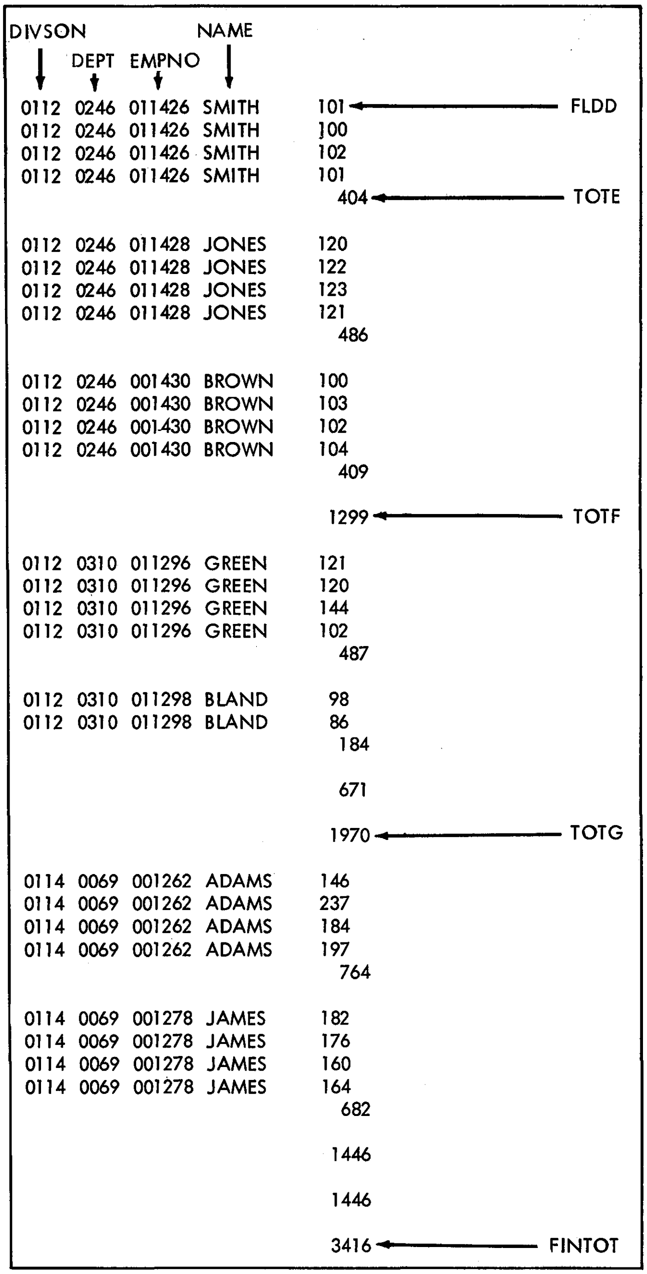
   c. The L1 in Output Indicators (columns 23-24) indicates that the line is to be printed only on a level-1 control break.

   d. The field to be printed (TOTE) is indicated under Field Name (columns 32-37).

   e. The Z in Zero Suppress (column 38) indicates that zeros to the left of significant digits are not to be printed.

   f. The B in Blank After (column 39) causes the main-storage positions containing field TOTE to be set to zeros after the total is printed. This is done so the total for one group is not added to the total of the previous group.

The specifications to print the second and third control levels are essentially the same as those for the first level.



```
DIVSON              NAME
 |   DEPT  EMPNO     |
 ↓    ↓      ↓       ↓
 0112 0246 011426 SMITH    101 ←—————————— FLDD
 0112 0246 011426 SMITH    100
 0112 0246 011426 SMITH    102
 0112 0246 011426 SMITH    101
                             404 ←———————— TOTE

 0112 0246 011428 JONES    120
 0112 0246 011428 JONES    122
 0112 0246 011428 JONES    123
 0112 0246 011428 JONES    121
                             486

 0112 0246 001430 BROWN    100
 0112 0246 001430 BROWN    103
 0112 0246 001430 BROWN    102
 0112 0246 001430 BROWN    104
                             409

                           1299 ←———————— TOTF

 0112 0310 011296 GREEN    121
 0112 0310 011296 GREEN    120
 0112 0310 011296 GREEN    144
 0112 0310 011296 GREEN    102
                             487

 0112 0310 011298 BLAND     98
 0112 0310 011298 BLAND     86
                             184

                             671

                           1970 ←———————— TOTG

 0114 0069 001262 ADAMS    146
 0114 0069 001262 ADAMS    237
 0114 0069 001262 ADAMS    184
 0114 0069 001262 ADAMS    197
                             764

 0114 0069 001278 JAMES    182
 0114 0069 001278 JAMES    176
 0114 0069 001278 JAMES    160
 0114 0069 001278 JAMES    164
                             682

                            1446

                            1446

                            3416 ←——————— FINTOT
```

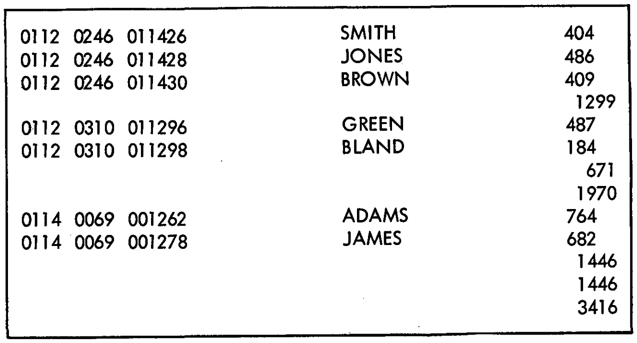
Figure 9. Detailed Printed Report

4. The specifications for printing the final total contain the code LR (Last Record) in Output Indicators (columns 23-24). In this example, the indicator LR is turned on automatically by the program when the last card of a file has been sensed. This indicator is used to cause the final total to be printed.

GROUP PRINTING

In group-printing operations only one line is printed for each group of detail cards. This line usually contains the control fields and the totals of the quantity fields.

An example of a group-printed report is shown in Figure 10.



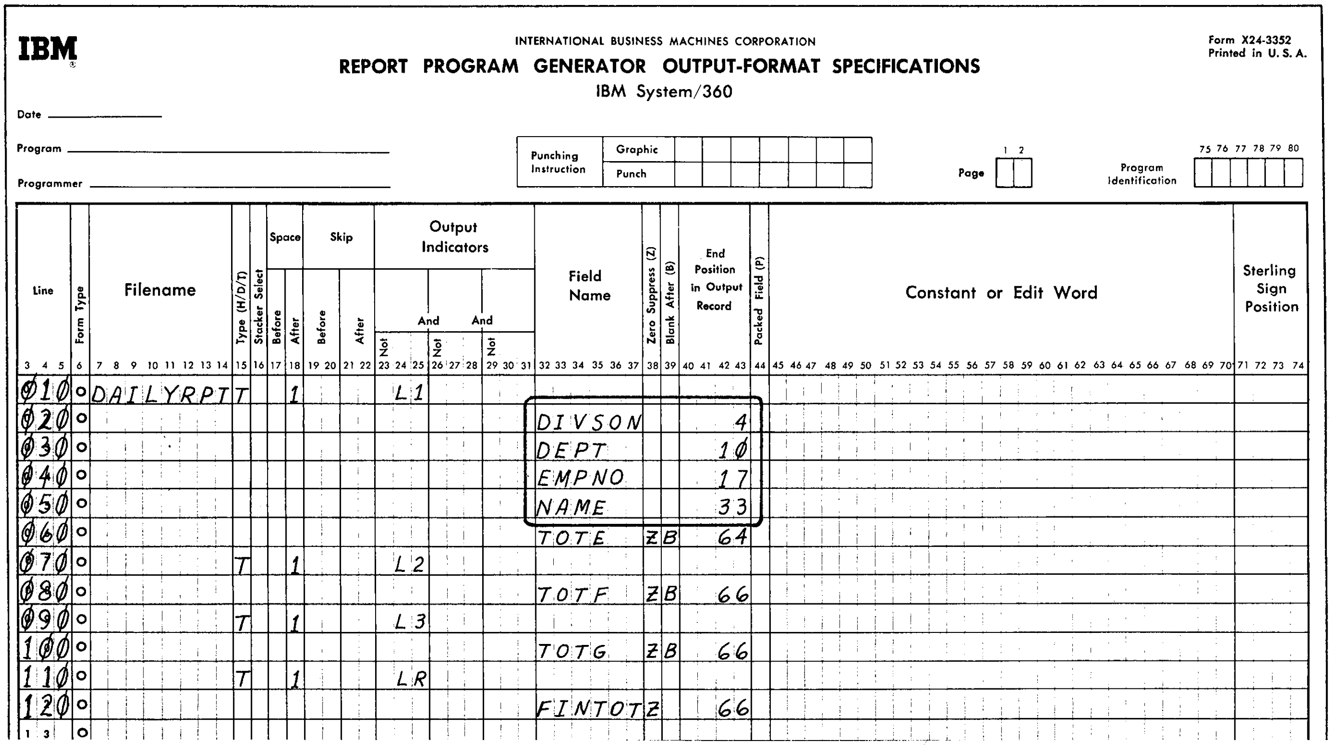
```
0112 0246 011426          SMITH          404
0112 0246 011428          JONES          486
0112 0246 011430          BROWN          409
                                           1299
0112 0310 011296          GREEN          487
0112 0310 011298          BLAND          184
                                            671
                                           1970
0114 0069 001262          ADAMS          764
0114 0069 001278          JAMES          682
                                           1446
                                           1446
                                           3416
```

Figure 10. Group-Printed Report

The detail-printed report specification in Figure 8 could be altered to provide a group-printed report as illustrated by the specifications in Figure 11.

GROUP INDICATION

In group-indication operations each detail card is processed; however, only the control fields that identify the specific detail card are printed. An example of a group-indication report is illustrated in Figure 12. In this example the employee name and number are printed for each control change. The fields DIVSON and DEPT are printed only when there is a control change for the division and department fields, respectively.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date ____________

Program ____________

Programmer ____________

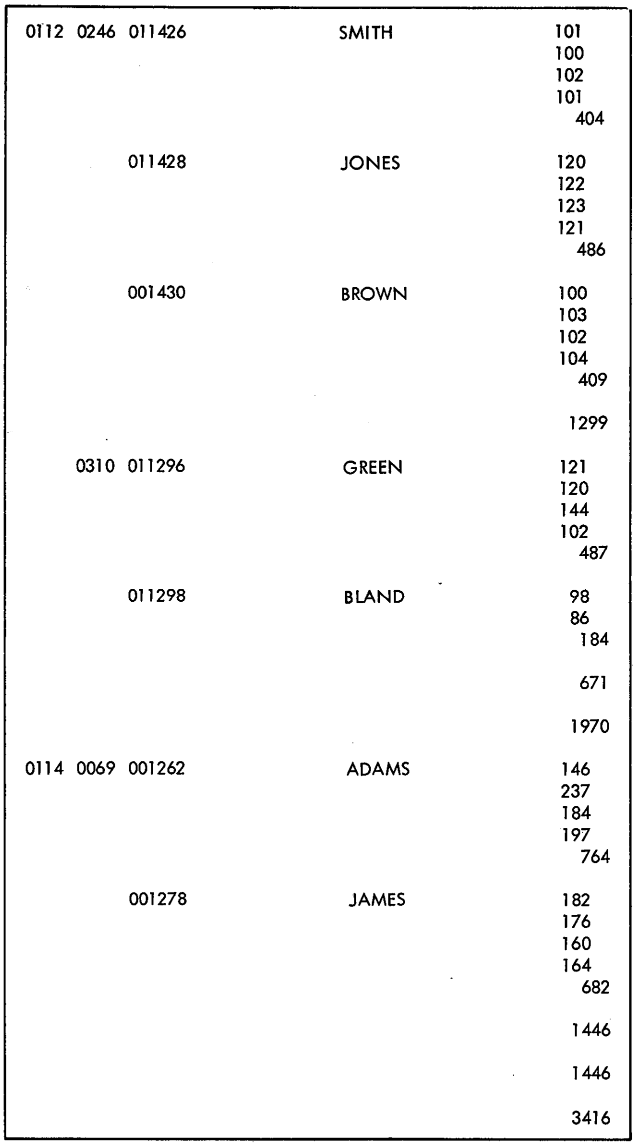
Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators Not		And Not		And Not		Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
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
Ø1Ø	O	DAILYRPT	T			1				L1											
Ø2Ø	O														DIVSON			4			
Ø3Ø	O														DEPT			1Ø			
Ø4Ø	O														EMPNO			17			
Ø5Ø	O														NAME			33			
Ø6Ø	O														TOTE	Z	B	64			
Ø7Ø	O		T			1				L2											
Ø8Ø	O														TOTF	Z	B	66			
Ø9Ø	O		T			1				L3											
1ØØ	O														TOTG	Z	B	66			
11Ø	O		T			1				LR											
12Ø	O														FINTOT	Z		66			
1 3	O																				

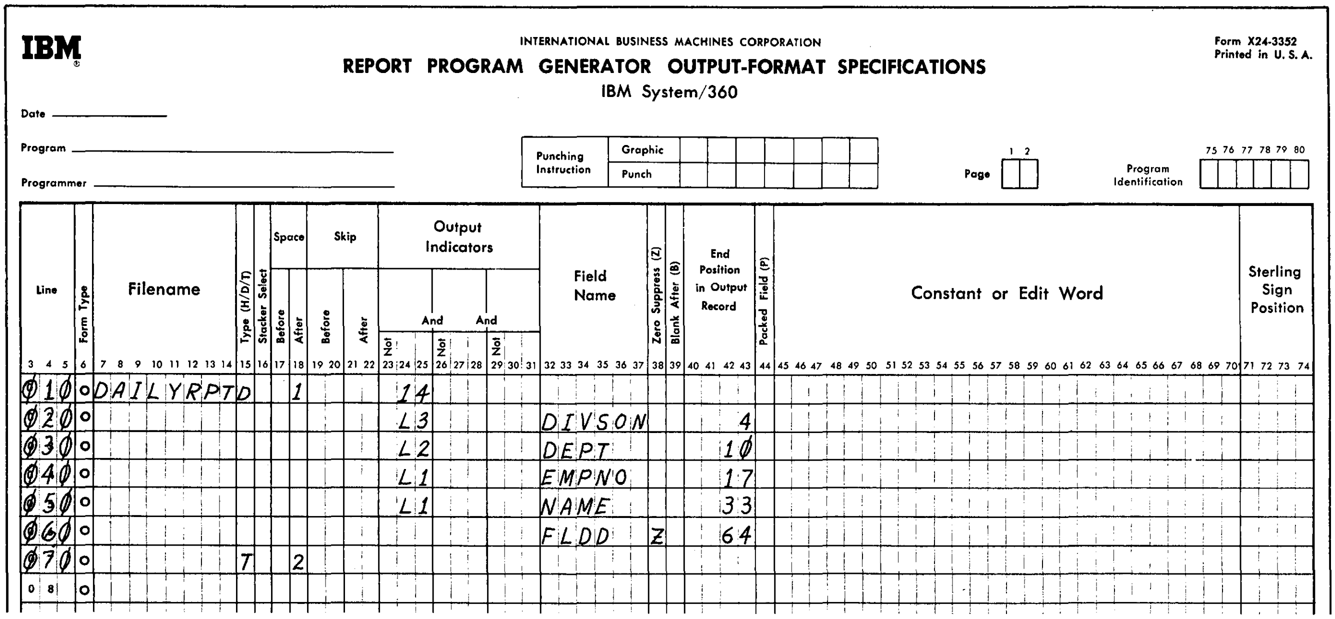
Figure 11. Specifying a Group-Printed Report



```
0112  0246  011426            SMITH          101
                                             100
                                             102
                                             101
                                               404

            011428            JONES          120
                                             122
                                             123
                                             121
                                               486

            001430            BROWN          100
                                             103
                                             102
                                             104
                                               409

                                              1299

      0310  011296            GREEN          121
                                             120
                                             144
                                             102
                                               487

            011298            BLAND           98
                                              86
                                               184

                                               671

                                              1970

0114  0069  001262            ADAMS          146
                                             237
                                             184
                                             197
                                               764

            001278            JAMES          182
                                             176
                                             160
                                             164
                                               682

                                              1446

                                              1446

                                              3416
```

Figure 12. Example of a Group-Indication Report

Figure 13 illustrates how the detail-printed report specifications in Figure 11 could be altered to specify a group-indication report.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date ____________

Program ____________

Programmer ____________

Punching Instruction: Graphic □□□□□□□ Punch □□□□□□□

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space		Skip		Output Indicators						Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
					Before	After	Before	After	Not		And Not		And Not								
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
Ø1Ø	O	DAILYRPT	D			1				14											
Ø2Ø	O									L3					DIVSON			4			
Ø3Ø	O									L2					DEPT			1Ø			
Ø4Ø	O									L1					EMPNO			17			
Ø5Ø	O									L1					NAME			33			
Ø6Ø	O														FLDD	Z		64			
Ø7Ø	O		T			2															
0 8	O																				

Figure 13. Specifying a Group-Indication Report

OVERFLOW PRINTING

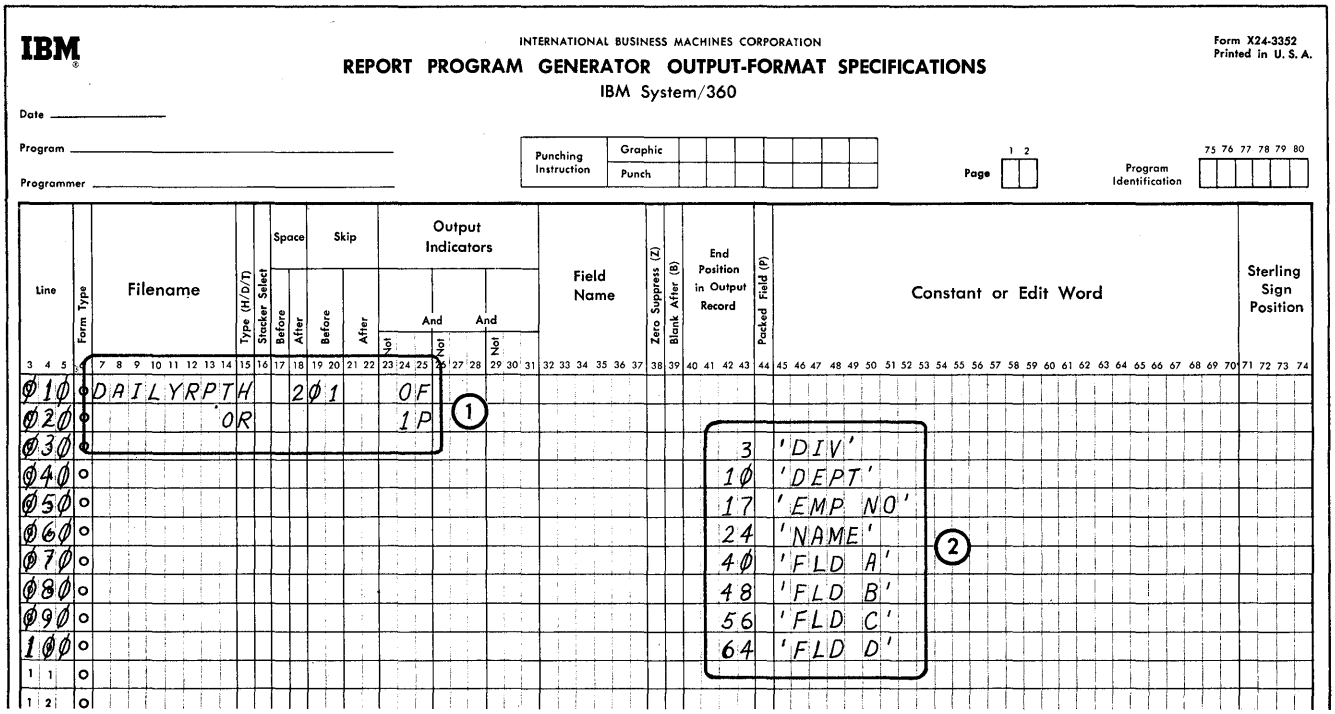
Overflow printing, another function used in preparing reports, can be performed in RPG. Overflow is the sensing of channel 12 in the printer carriage control tape.

Problem

This example shows the additional specifications necessary to print eight column headings at the top of each printed form. A heading-line is a line that contains information from an input record or contains constants.

Specifications

Figure 14 illustrates the output specifications required for this operation. The numbers in the following list refer to the circled items in Figure 14. The File Name (columns 7-14) of DAILYRPT is the same; adding overflow headings does not change it.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
Ø1Ø	o	DAILYRPT	H			2	Ø1		OF							
Ø2Ø	o	OR							1P							
Ø3Ø	o												3		'DIV'	
Ø4Ø	o												1Ø		'DEPT'	
Ø5Ø	o												17		'EMP NO'	
Ø6Ø	o												24		'NAME'	
Ø7Ø	o												4Ø		'FLD A'	
Ø8Ø	o												48		'FLD B'	
Ø9Ø	o												56		'FLD C'	
1ØØ	o												64		'FLD D'	
1 1	o															
1 2	o															

Figure 14 Printing Heading Lines

1. On the first line of the form, the H in H/D/T (column 15) indicates that the line to be printed is a heading line.

   The 2 in Space After (column 18) provides a blank line after each printed line.

   The 01 in Skip Before (columns 19-20) causes the form to skip to channel 01 in the carriage control tape. This positions the form for printing of the overflow heading line.

   The OF in Output Indicators (columns 23-24) causes the heading line to be printed each time there is a form overflow on the printer.

   On the second line of the form, the letters OR indicate a second condition can cause the heading line to print. The second condition, indicated by 1P (first page) in Output Indicators, causes the heading line to be printed on the first page. This condition is necessary to print the heading on the first page of the report, because the overflow condition does not occur until after the first page of the report has been printed.

2. The entries on lines 3-10 of Figure 14 specify the actual information or constants to be printed on the report. The actual information is contained within single quote signs.

TESTING FOR ZERO, PLUS, AND MINUS BALANCE

Specifications are executed in the object program in the same sequence in which they are written on the specification form unless a different sequence has been specified.

If specifications had to be followed sequentially in a fixed pattern, a program would follow a single path of operation without any possibility of dealing with predefined exceptions to the procedure, and without the ability to choose a predefined alternative to the procedure, based upon conditions encountered during the processing of the program.

For example, assume that a program has been written containing ten specifications that cause a number of operations to be performed upon a series of quantities. If the first five specifications develop meaningless results when processed with quantities of zero, the processing time of the object program can be reduced if the first five specifications are bypassed whenever the quantity to be processed is zero.

A specification in the RPG program can be used to evaluate a quantity and, depending upon the value of that quantity, direct the program to some other specification.

Input Specifications

Three types of tests can be made on the Input Specification sheet:

1. Testing an input field to determine if it contains a plus value.
2. Testing an input field to determine if it contains a minus value.
3. Testing an input field to determine if it is blank or is punched with zero.

Calculation Specifications

Three types of tests can be made on the Calculation Specification sheet: tests to determine whether the result of a calculation is plus, minus, or zero.

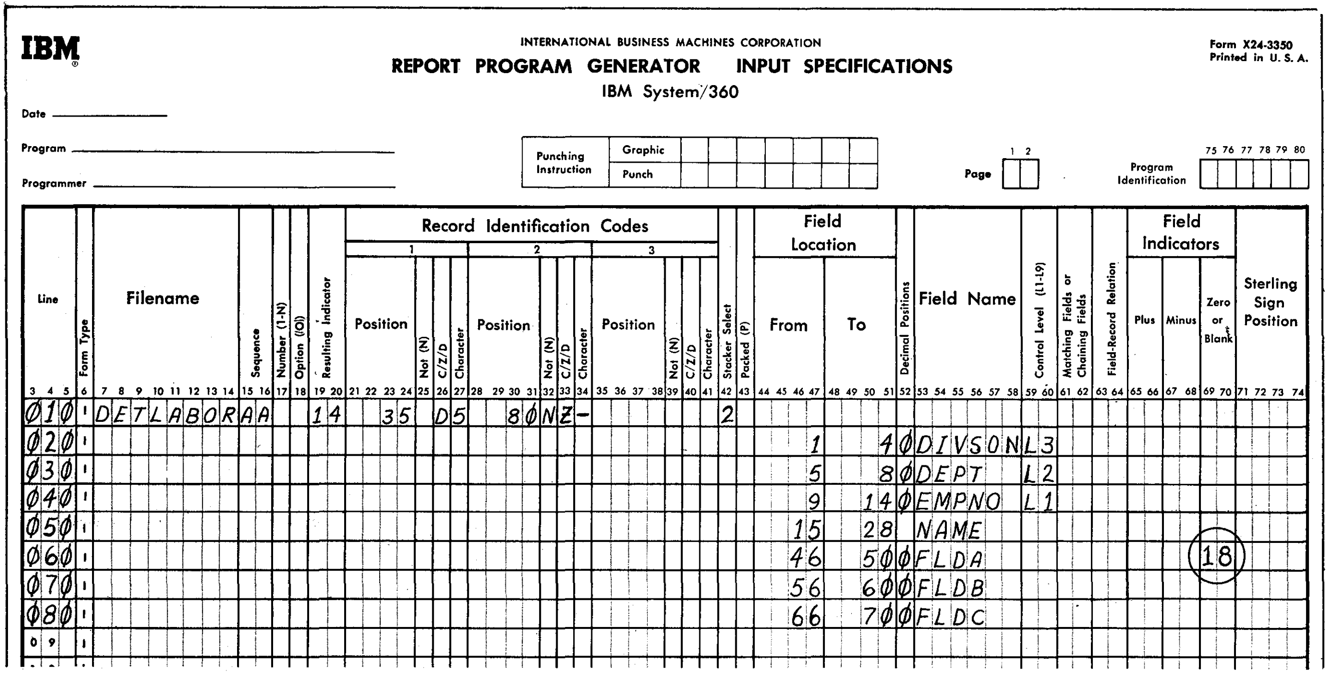
The program also can compare two fields and can test the result to determine if the content of one field is greater than, smaller than, or equal to that of the other field.

Figure 15 and 16 illustrate a test for a zero balance, and Figure 17 shows the specifications for a test for a minus balance.

Figure 15 illustrates a typical test of the content of an input field. If FLDA (line 6) contains zeros, it is unnecessary to perform some of the operations entered on the Calculation Specification sheet. To make this test, the programmer places a number from 01 to 99 in Field Indicators: Zero or Blank (columns 69-70). In this example the number is 18. If a card read into the system contains zeros in columns 46-50 (FLDA), indicator 18 is turned on.

Turning on an indicator means a special condition has occurred and the program must consider this condition during the processing of calculation specifications and/or output specifications. This indicator condition is no different from having a resulting indicator turned on by a specific record identification code from an input card.

For example, the entries circled in Figure 16 are the additional specifications that bypass the calculation upon detail cards if the value in FLDA is zero or blank.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date

Program

Programmer

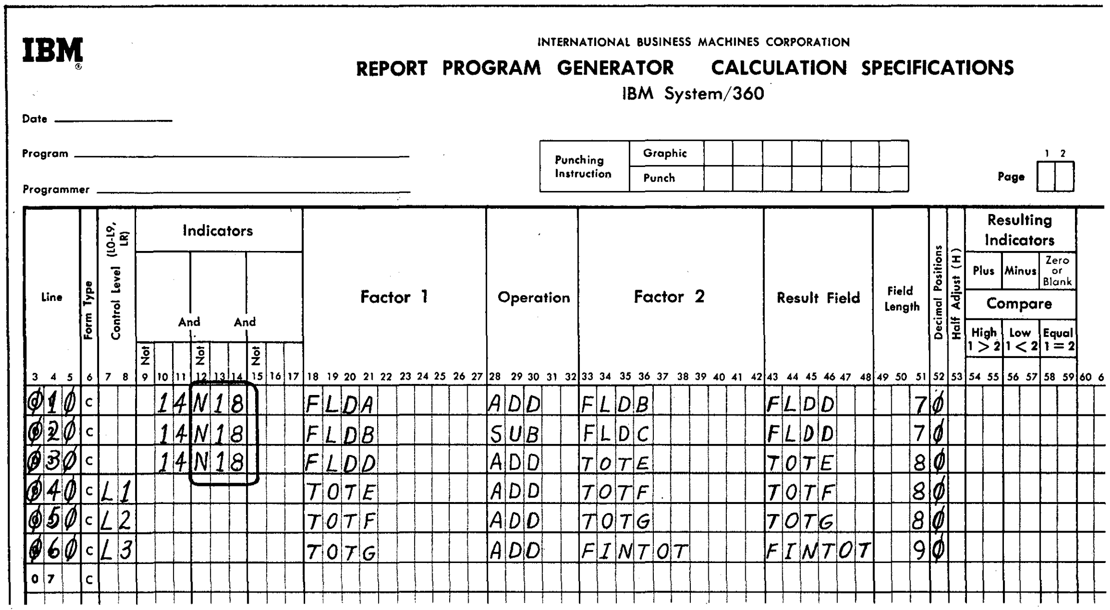
Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Resulting Indicator (19-20)	1 Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	2 Position (28-31)	Not (N) (32)	C/Z/D (33)	Character (34)	3 Position (35-38)	Not (N) (39)	C/Z/D (40)	Character (41)	Stacker Select (42)	Packed (P) (43)	Field Location From (44-47)	Field Location To (48-51)	Decimal Positions (52)	Field Name (53-58)	Control Level (L1-L9) (59-60)	Matching Fields or Chaining Fields (61-62)	Field-Record Relation (63-64)	Field Indicators Plus (65-66)	Field Indicators Minus (67-68)	Field Indicators Zero or Blank (69-70)	Sterling Sign Position (71-74)
∅1∅	I	DETLABOR	AA			14	35		D	5	8∅	N	Z	-					2												
∅2∅	I																				1	4	∅	DIVSON	L3						
∅3∅	I																				5	8	∅	DEPT	L2						
∅4∅	I																				9	14	∅	EMPNO	L1						
∅5∅	I																				15	28		NAME							
∅6∅	I																				46	5∅	∅	FLDA						18	
∅7∅	I																				56	6∅	∅	FLDB							
∅8∅	I																				66	7∅	∅	FLDC							
09	I																														

Figure 15. Specifying a Test for a Zero-Balance



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Line	Form Type	Control Level (L0-L9, LR)	Indicators (Not, And, Not, And, Not)	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank; Compare: High 1>2 / Low 1<2 / Equal 1=2
Ø1Ø	C		14N18	FLDA	ADD	FLDB	FLDD	7	Ø		
Ø2Ø	C		14N18	FLDB	SUB	FLDC	FLDD	7	Ø		
Ø3Ø	C		14N18	FLDD	ADD	TOTE	TOTE	8	Ø		
Ø4Ø	C	L1		TOTE	ADD	TOTF	TOTF	8	Ø		
Ø5Ø	C	L2		TOTF	ADD	TOTG	TOTG	8	Ø		
Ø6Ø	C	L3		TOTG	ADD	FINTOT	FINTOT	9	Ø		
07	C										

Figure 16. Test for a Zero-Balance

The specifications 14N18 in Indicators (columns 10-14) mean that the calculation will be performed if indicator 14 is on and indicator 18 is off. (The N in N18 stands for Not.) As now written on the Calculation Specification sheet in Figure 16, the detail calculations are not performed if the value of FLDA from the input card is zero (Figure 15).

The same indicator specification, N18, could also be used on the Output-Format Specification sheet to prevent the printing of detail cards when FLDA is zero (if that was a requirement of the program).

USING RESULTING INDICATORS

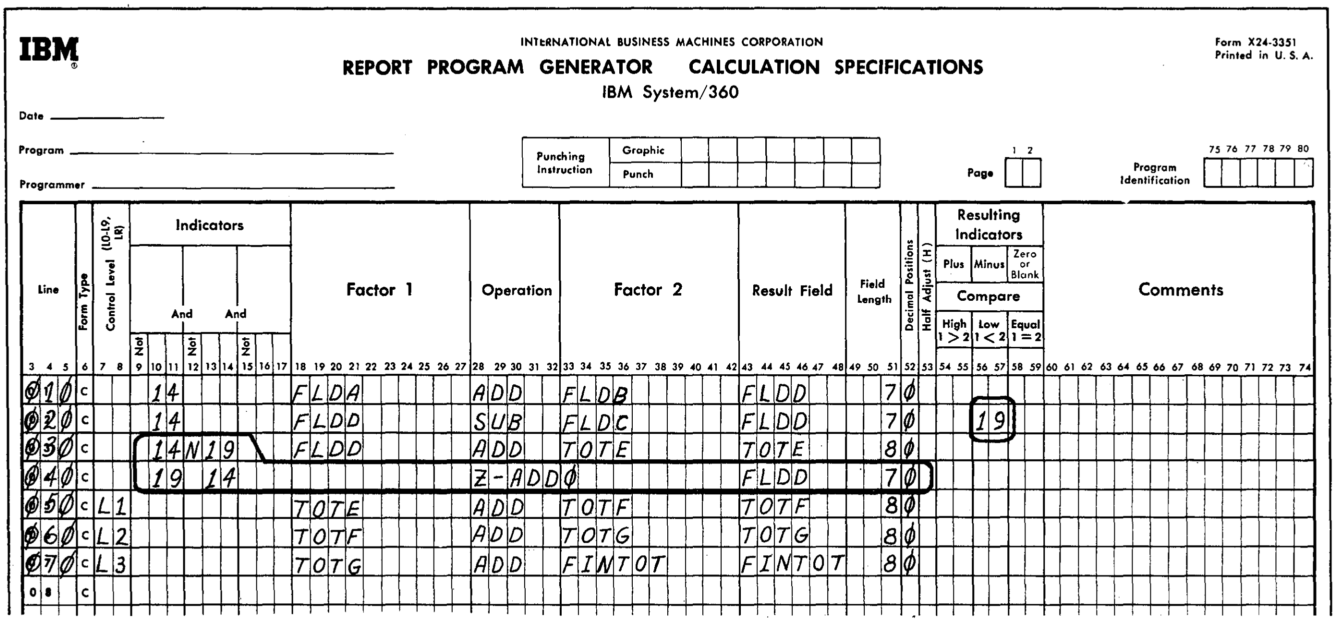
Problem

The calculation specifications in this example (see Figure 17) illustrate the use of a resulting indicator to test for a minus balance. The result of the test can be used to bypass some specifications and to process other specifications only when the condition tested for is present.

Specifications

The specifications for this example are shown in Figure 17. The program logic for this example is shown in Figure 18.

Line 1 specifies that FLDB is to be added to FLDA and that the result is to be placed in FLDD. Line 2 specifies that FLDC is to be subtracted from FLDD, that the result is to be placed in FLDD, and that FLDD is to be tested to determine if the result is minus. In this example it is assumed that if a minus balance occurs, the calculation has no meaning. Therefore, if the result is minus, two things must be accomplished:



IBM INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date ______

Program ______

Programmer ______

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Control Level (L0-L9, LR)	Indicators: Not	Indicators	And: Not	And	And: Not	And	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Compare: High 1>2	Minus / Low 1<2	Zero or Blank / Equal 1=2	Comments
3 4 5	6	7 8	9	10 11	12	13 14	15	16 17	18-27	28-32	33-42	43-48	49-51	52	53	54 55	56 57	58 59	60-74
Ø1Ø	C			14					FLDA	ADD	FLDB	FLDD	7	Ø					
Ø2Ø	C			14					FLDD	SUB	FLDC	FLDD	7	Ø			19		
Ø3Ø	C			14	N	19			FLDD	ADD	TOTE	TOTE	8	Ø					
Ø4Ø	C			19		14				Z-ADD	Ø	FLDD	7	Ø					
Ø5Ø	C	L1							TOTE	ADD	TOTF	TOTF	8	Ø					
Ø6Ø	C	L2							TOTF	ADD	TOTG	TOTG	8	Ø					
Ø7Ø	C	L3							TOTG	ADD	FINTOT	FINTOT	8	Ø					
0 8	C																		

Figure 17. Testing for a Minus Condition

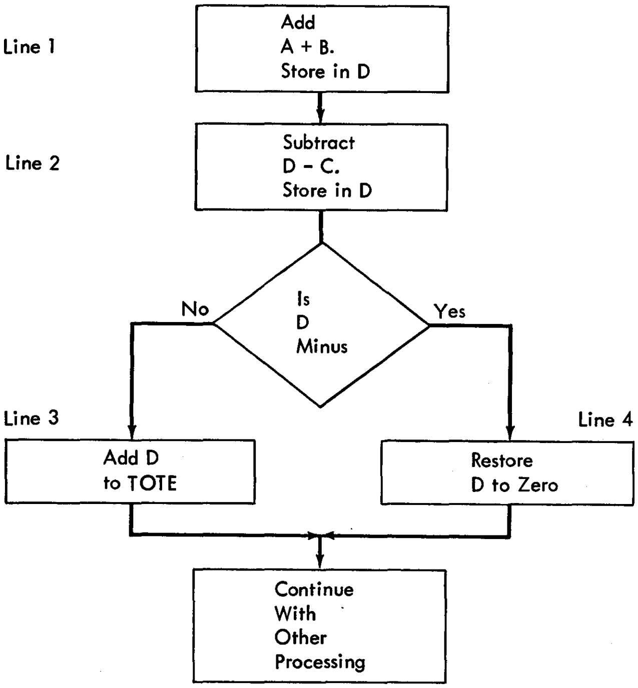


Figure 18. Testing Indicators to Govern Processing

1. The result must not be added into field TOTE.

2. The contents of field FLDD must be reset to zeros. (This step might be required if FLDD were used in a subsequent step and the minus balance remaining would give incorrect results.)

This is accomplished on the specification sheet (Figure 17) by placing an indicator code (19) in Resulting Indicators: Minus on specification-line 2. Indicator 19 will be turned on for a minus condition.

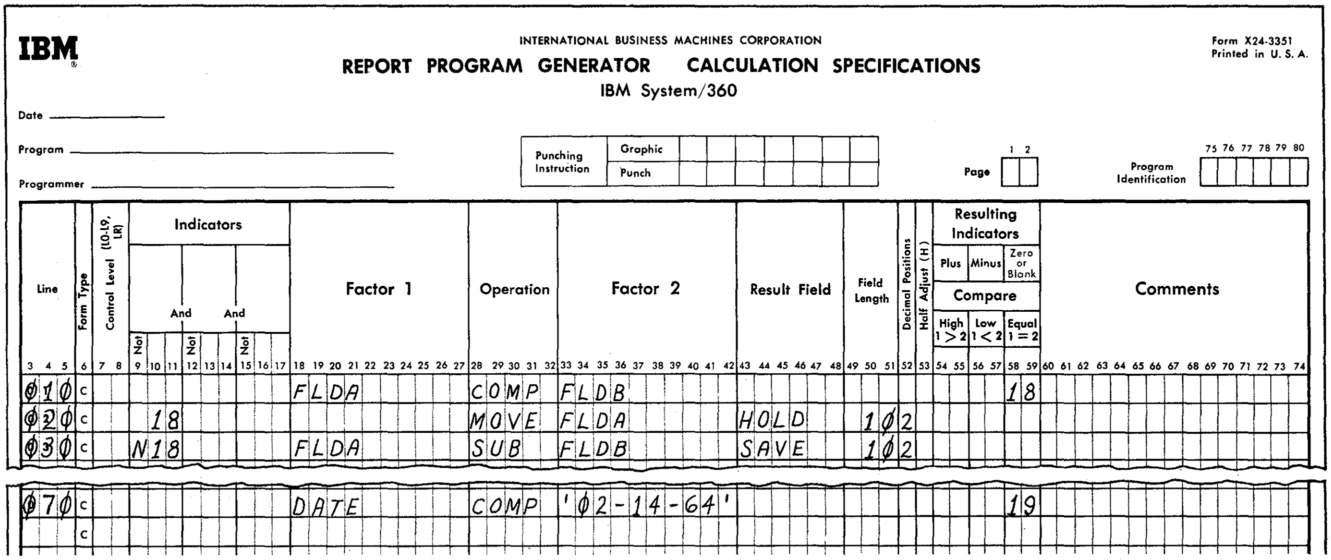
The function of adding TOTE to FLDD is specified on line 3. It is accomplished only if there is a no-minus condition resulting from the test on line 2. The specifications on line 4 cause FLDD to be reset to zeros only on a minus condition.

The 0 in Factor 2 on line 4 is known as a literal and is used to set FLDD to zeros. This is accomplished by the zero and add (Z-ADD) operation. A literal is the actual value to be used in a calculation rather than the name of the location of the data to be used. The remaining specifications on the form in Figure 17 are the same as those from previous examples.

By using indicator 19, it is possible to suppress detail-card printing when D - C results in a minus balance.

COMPARISON OF TWO FIELDS

The calculation specifications in this example illustrate the ability to compare two fields and govern processing according to the result of the comparison. The comparison may be tested for a high, low, or equal condition. The result of the test can then be used to bypass some specifi-



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page

Program Identification

Line	Form Type	Control Level (L0-L9, LR)	Indicators (Not)	And (Not)	And (Not)	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Compare High 1>2	Minus / Low 1<2	Zero or Blank / Equal 1=2	Comments
Ø1Ø	C					FLDA	COMP	FLDB							18	
Ø2Ø	C		18				MOVE	FLDA	HOLD	1Ø	2					
Ø3Ø	C		N18			FLDA	SUB	FLDB	SAVE	1Ø	2					
Ø7Ø	C					DATE	COMP	'Ø2-14-64'							19	
	C															

Figure 19. Specifying a Comparison

cations or to process other specifications only when a particular condition is present.

Specifications

The calculation specifications for this example are shown in Figure 19. The program logic for the example is shown in Figure 20.

Line 1 specifies that FLDA is compared to FLDB. If the two fields are equal, indicator 18 is turned on, and FLDA is moved to HOLD (line 2). If the two fields are not equal (line 3), FLDB is subtracted from FLDA and the result is placed in SAVE.

A literal may also be used in a comparison specification. In the lower half of Figure 19, the contents of the input field, DATE, are compared against 02-14-64, and indicator 19 is turned on if they are equal.

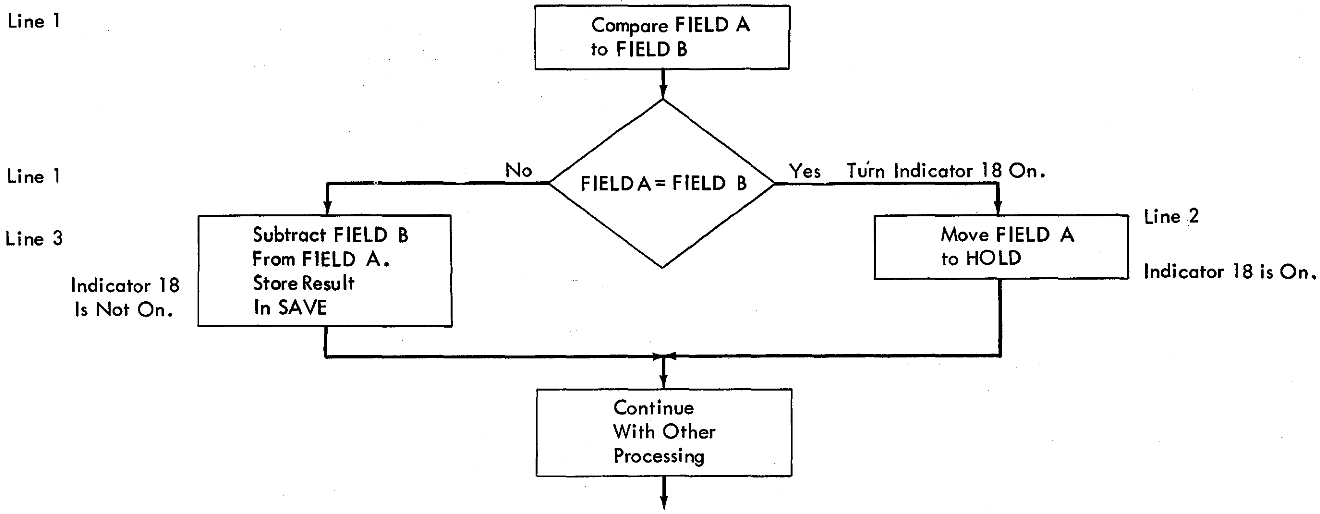


Figure 20. Testing a Comparison

MULTIPLICATION AND DIVISION

Multiply and divide operations are easily accomplished with RPG. Moreover, two problems associated with these functions (decimal-point alignment and half-adjusting) are easily specified.

Problem

This example illustrates an inventory card (Figure 21) that must have the quantity multiplied by the price to develop a total cost on a weekly basis. The price of the part is also updated each week, so that the price for the following week reflects the average fabrication costs and scrap losses on a year-to-date basis.

Specifications

The specifications to accomplish these functions are shown in Figures 22 and 23. Figure 22 shows the input specifications for all five fields. The programmer must be certain of the exact size of each field and the number of decimal places within each field. The fields in this example are:

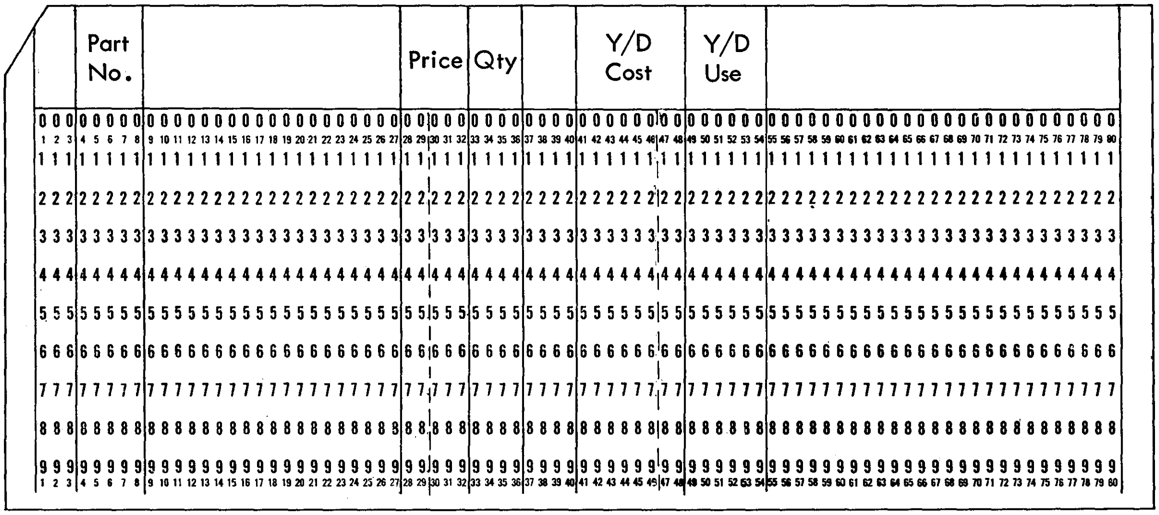
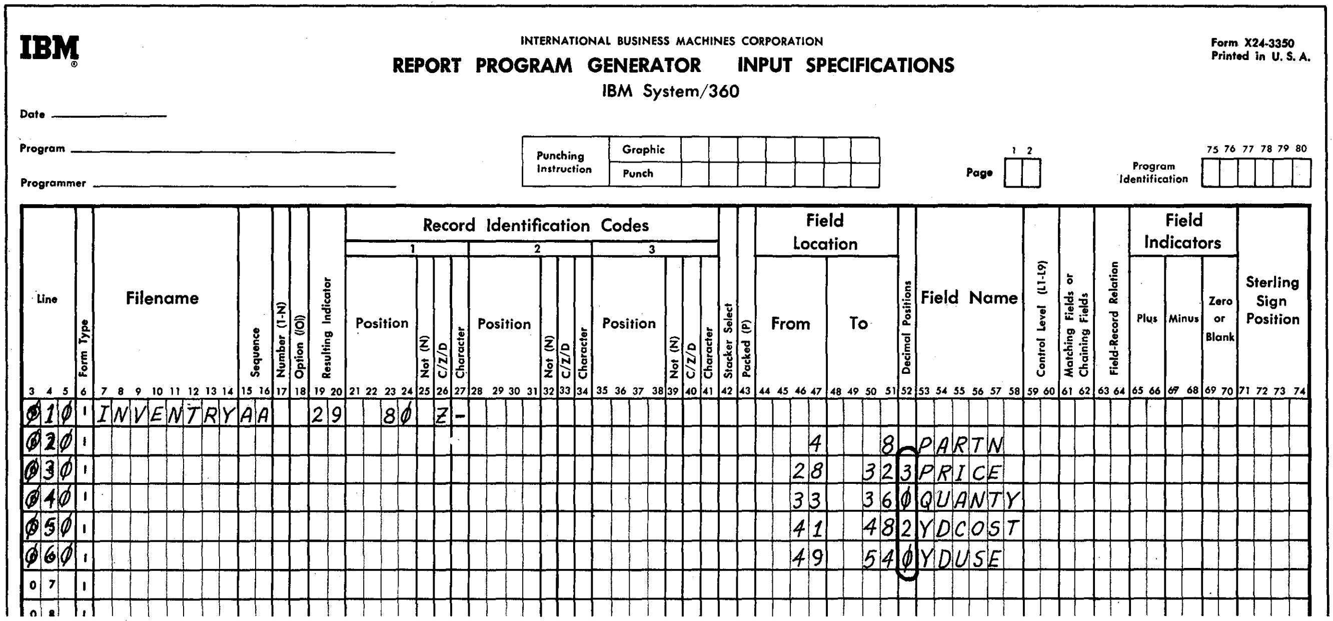


Figure 21. Inventory Card



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U.S.A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page

Program Identification

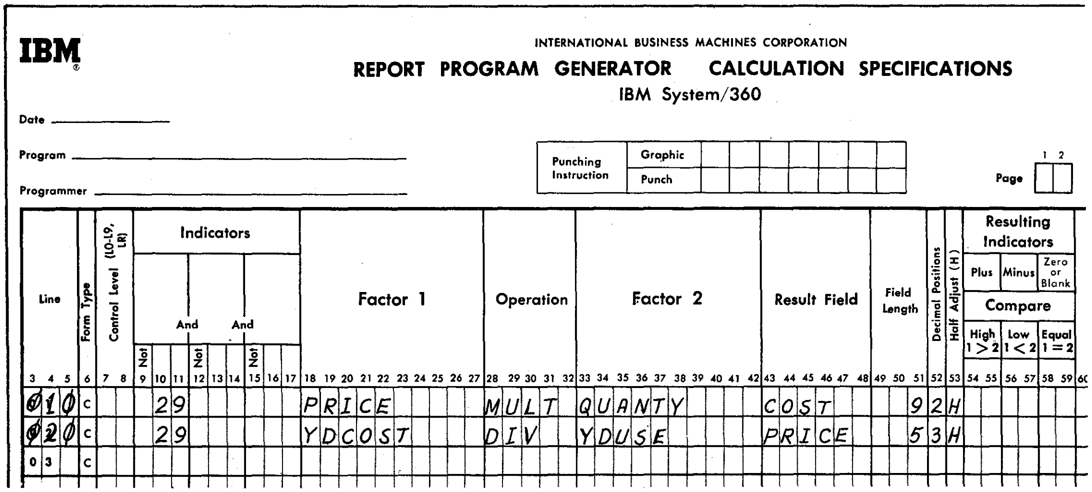
Line	Form Type	Filename	Resulting Indicator	Record Identification Codes 1 Position	C/Z/D	Character	Field Location From	To	Decimal Positions	Field Name
Ø1Ø	I	INVENTRYAA	29	8Ø	Z	-				
Ø2Ø	I						4	8		PARTN
Ø3Ø	I						28	32	3	PRICE
Ø4Ø	I						33	36	Ø	QUANTY
Ø5Ø	I						41	48	2	YDCOST
Ø6Ø	I						49	54	Ø	YDUSE
07	I									
08	I									

Figure 22. Input Specifications

		No. of Decimal Positions
Part Number	xxxxx	None (alphabetic)
Price	xx.xxx	3
Quantity	xxxx.	0
Y/D Cost	xxxxxx.xx	2
Y/D Usage	xxxxxx.	0

As shown in Figure 22, the decimal positions of each field are entered under Decimal Positions on the Input Specification sheet.

The calculation specifications for this example are shown in Figure 23. PRICE is multiplied by QUANTY and the result is placed in a field called COST. The field length of COST is specified as 9. The result field is to have two decimal positions and to be half-adjusted as specified by the H in Half-Adjust (column 53). The following is an example of the arithmetic of this operation, using actual values:



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)
Ø1Ø	C	29	PRICE	MULT	QUANTY	COST	9	2	H
Ø2Ø	C	29	YDCOST	DIV	YDUSE	PRICE	5	3	H
03	C								

Figure 23. Calculation Specifications

```
Price           1.213
Quantity        x 216
                7 278
               12 13
              242 6
              262.008
```

The result field was specified for two decimal positions; therefore, the half-adjustment is made to the digit 8 in the units position. Half-adjustment is always made to the position to the right of the last position retained as part of the result, as follows:

```
262.008
+     5 half-adjust
262.013
```

The result, 262.01, is stored as the contents of COST. The position that was half-adjusted is dropped.

The second line of the Calculation Specification sheet provides the specifications for dividing year-to-date costs by year-to-date usage. The result is placed in a field called PRICE. The field length of PRICE is specified as 5, with three decimal positions. The result is to be half-adjusted, as specified by the H in Half-Adjust. An example of the arithmetic of this operation, using actual values, follows. Year-to-date COST divided by usage equals PRICE.

```
                    1.3419  (price)
(usage)    1296) 1739.23xx  (year-to-date
                            cost)
```

The result field was specified for three decimal places. Therefore, the half-adjustment is made to the digit 9 in the units position. This is the position to the right of the last position retained as part of the result, as follows:

```
1.3419
+    5 half-adjust
1.3424
```

The result, 1.342, is stored as the contents of PRICE. The position that was half-adjusted is dropped.

SEQUENCE-CHECKING

Two types of sequence-checking functions can be performed with RPG:

1. Checking the sequence of different record-types within a control group.

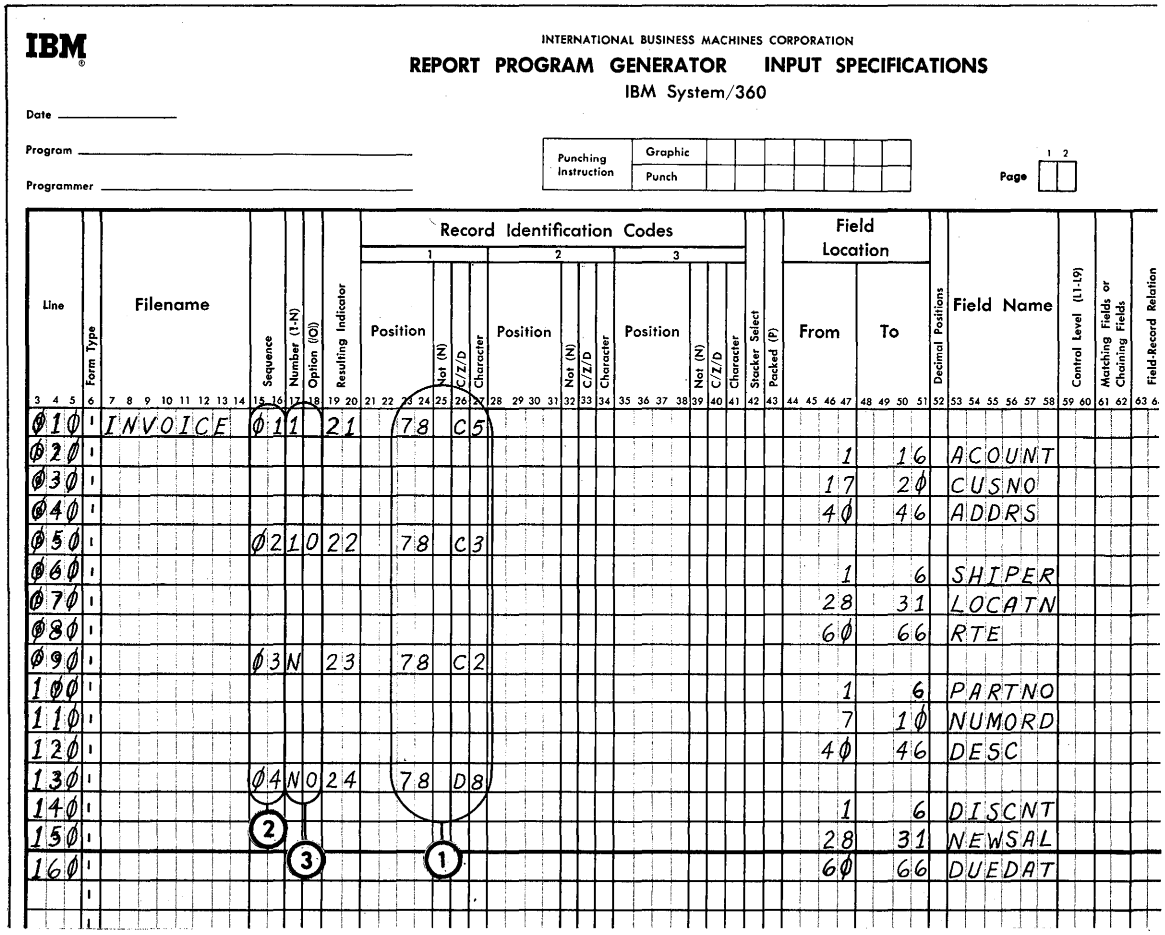
2. Checking the sequence of control groups. (See the section Using the Matching Fields Specification for Sequence-Checking.)

Sequence-Checking of Different Record Types within a Control Group

The application consists of updating an invoice file, which, in this case, contains from two to four of the following types for each part number:

Card	Code
Master Customer	5 in col. 78
Shipped Via	3 in col. 78
Part Number	2 in col. 78
Comments	8 in col. 78

Figure 24 illustrates the specifications required to check the sequence of a group of four card-types. The numbers on the form refer to the following numbers:



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Date

Program

Programmer

Punching Instruction	Graphic	Punch

Page 1 2

							Record Identification Codes														Field Location						
							1				2				3												
Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation
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
Ø1Ø	I	INVOICE	Ø1	1		21	78		C	5																	
Ø2Ø	I																				1	16		ACOUNT			
Ø3Ø	I																				17	2Ø		CUSNO			
Ø4Ø	I																				4Ø	46		ADDRS			
Ø5Ø	I		Ø2	1	O	22	78		C	3																	
Ø6Ø	I																				1	6		SHIPER			
Ø7Ø	I																				28	31		LOCATN			
Ø8Ø	I																				6Ø	66		RTE			
Ø9Ø	I		Ø3	N		23	78		C	2																	
1ØØ	I																				1	6		PARTNO			
11Ø	I																				7	1Ø		NUMORD			
12Ø	I																				4Ø	46		DESC			
13Ø	I		Ø4	N	O	24	78		D	8																	
14Ø	I																				1	6		DISCNT			
15Ø	I																				28	31		NEWSAL			
16Ø	I																				6Ø	66		DUEDAT			

Figure 24. Input Specifications

1. The record-identification codes for the four cards are specified in the same manner as in previous examples. The C in C/Z/D indicates that the entire character punched in the card is examined to establish the record identification code.

   In the specifications for the last card-type (specification-line 13 on the form), a D is written in C/Z/D because there is a possibility that some of these card-types may have a zone punch. The D specifies that only the digit punches in the card are examined to identify the card-type. Thus, zone punches that could result in unequal comparisons are ignored.

2. The sequence established for the file is determined by the sequence in which the specifications for each card-type are written on the form and by the numbers placed in Sequence. In this example, the digits 01, 02, 03, and 04 are used. The numbers assigned must begin with 01 in each file.

   Alphabetic characters under Sequence in preceding examples indicate that no sequence-checking is to take place. Alphabetic specifications must always be written before numeric specifications for any one file.

   These specifications are all that are required to cause the object program to perform a sequence-check of the various record-types. If a sequence-check error is detected, the object program is halted.

   Two additional specifications, Number and Option, are used in these types of applications.

3. If a numeric specification is provided in Sequence, a specification must be provided in Number.

On the first two record identification lines, the number 1 in Number indicates that no more than one record of that type must be present in each group. In this example, only one master card and no more than one shipped-via card for each invoice must be present. If there is more than one of either of these cards, the program recognizes an error in the input file.

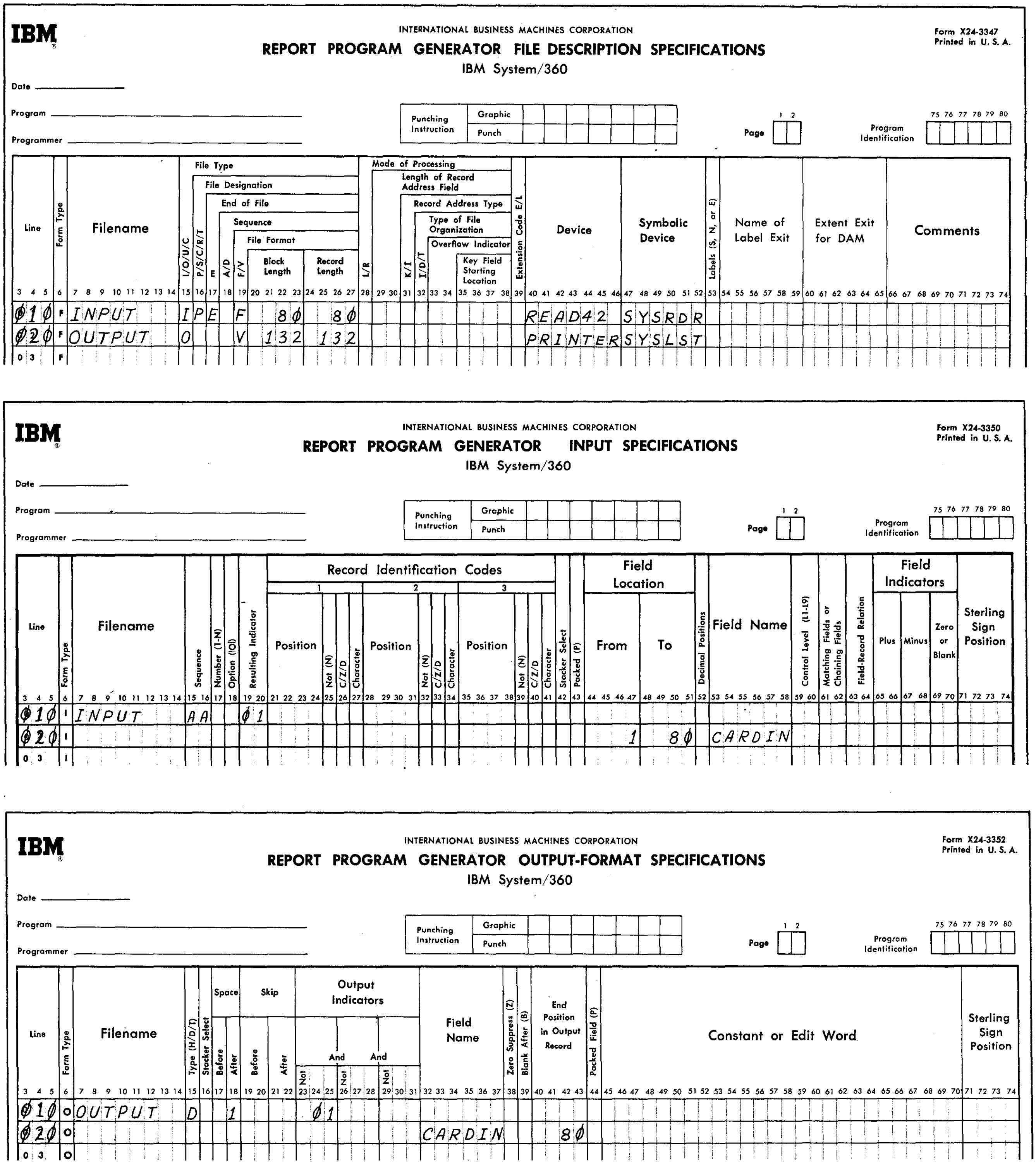
The letter N in Number of the specifications for the last two record-types means that multiple card-types for each part number may be present. In this case, multiple cards for part number and comments may be present.

The letter O in Option in the specifications means that the records are optional. That is, a record may or may not be present.

If the letter O was not specified, it means that the particular record must be present. This requirement applies only if Sequence has been specified as numeric.

CORRELATION OF THE RPG SPECIFICATION SHEETS

Figure 25 shows a file-to-file sample program. It illustrates the relationships of the RPG specification sheets. (This type of report is often called an 80/80 listing).



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date ____
Program ____
Programmer ____

Punching Instruction: Graphic / Punch

Page 1 2 | Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	File Type I/O/U/C (15)	File Designation P/S/C/R/T (16)	End of File E (17)	Sequence A/D (18)	File Format F/V (19)	Block Length (20-23)	Record Length (24-27)	Mode of Processing L/R (28)	Length of Record Address Field (29-30)	Record Address Type K/I (31)	Type of File Organization I/D/T (32)	Overflow Indicator (33-34)	Key Field Starting Location (35-38)	Extension Code E/L (39)	Device (40-46)	Symbolic Device (47-52)	Labels (S, N, or E) (53)	Name of Label Exit (54-59)	Extent Exit for DAM (60-65)	Comments (66-74)
Ø1Ø	F	INPUT	I	P	E		F	8Ø	8Ø								READ42	SYSRDR				
Ø2Ø	F	OUTPUT	O				V	132	132								PRINTER	SYSLST				
0 3	F																					

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date ____
Program ____
Programmer ____

Punching Instruction: Graphic / Punch

Page 1 2 | Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Resulting Indicator (19-20)	Record Identification Codes 1 Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	2 Position (28-31)	Not (N) (32)	C/Z/D (33)	Character (34)	3 Position (35-38)	Not (N) (39)	C/Z/D (40)	Character (41)	Stacker Select (42)	Packed (P) (43)	Field Location From (44-47)	To (48-51)	Decimal Positions (52)	Field Name (53-58)	Control Level (L1-L9) (59-60)	Matching Fields or Chaining Fields (61-62)	Field-Record Relation (63-64)	Field Indicators Plus (65-66)	Minus (67-68)	Zero or Blank (69-70)	Sterling Sign Position (71-74)
Ø1Ø	I	INPUT	AA			Ø1																									
Ø2Ø	I																				1	8Ø		CARDIN							
0 3	I																														

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date ____
Program ____
Programmer ____

Punching Instruction: Graphic / Punch

Page 1 2 | Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators Not (23)	(24-25)	And Not (26)	(27-28)	And Not (29)	(30-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)	End Position in Output Record (40-43)	Packed Field (P) (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
Ø1Ø	O	OUTPUT	D			1				Ø1											
Ø2Ø	O														CARDIN			8Ø			
0 3	O																				

Figure 25. File-to-File Program

Assume the contents of an input file are to be transferred to another file. In this example the input file is a card file, and contents of the cards are to be printed. Three specifications sheets are required for this program: File Description, Input, and Output-Format.

FILE DESCRIPTION SHEET

The two files are described on this sheet. The card-input file is assigned the name INPUT, and the printed output file is named OUTPUT. Page and line sequence numbers are entered in columns 1-5. An F in column 6 indicates that each entry is a File Description entry. The file names are entered in columns 7-14 (Filename). Column 15 contains an I or an O to indicate whether the file has an input or an output function.

Column 16 of the input file on the File Description Specification sheet contains a P because the file described is the primary input file for the job; the E in column 17 indicates that the end-of-file condition for this file occurs when this file is depleted.

Column 19 contains an F and a V to indicate that the file formats are fixed-length and variable-length respectively. (See Variable-Length Records.)

Block length (columns 20-23) is 80 for the input file because each card is a block of data. Record length (columns 24-27) is also 80 for the input file because each card is an unblocked record. For the output file the block length and the record length is 132, which is the maximum length of a printer line. The program identification is entered into columns 75-80.

If the input file was read in by an IBM 1442 Card Read-Punch, the code in Device (columns 40-46) would be READ42. The output printer would have a device code of PRINTER.

The symbolic device codes for the two files would be SYSRDR and SYSLST.

INPUT SPECIFICATION SHEET

The Input Specification sheet also has the program identification entered in columns 75-80 and the page and line numbers in columns 1-5. An I in column 6 of each card indicates an Input Specification Entry.

The filename is again entered into columns 7-14. Columns 15-16 contain the sequence code AA. Indicator 01, entered into columns 19-20, will be on throughout the job to show that records from its associated file are being processed. The second line describes the field name CARDIN. The field consists of card columns 1-80.

OUTPUT-FORMAT SPECIFICATION SHEET

On the Output-Format Specification sheet, the filename of the output file is entered in columns 7-14. The D in column 15 indicates that each line printed in this file is a detail line. A single space after printing is specified by the entry in column 18. Indicator 01 from the Input Specification sheet is specified in columns 24-25. When indicator 01 is on, a record will be printed.

The second output line has the name of the field to be written in the output record entered in columns 32-37. Data from the field labeled CARDIN will be printed (end position of 80) in the output record as specified in columns 40-43.

SUMMARY

This completes the general description of some of the functions that can be performed with RPG. Some of the fields of the specifications sheets were not explained and some additional operations that can be performed with RPG remain to be described. At this point, the reader should be able to determine the scope of the RPG program.

More specific information about each specification sheet is contained under RPG Specification Sheets.

Each object program generated by RPG uses the same general logic, and for each record to be processed the program goes through the same general cycle of operations. Within that cycle, there are two different instances in time when operations specified on the Calculation and Output-Format Specification sheets are performed. These instances are called detail and total time.

For the illustration of this concept, a generalized flowchart of an RPG object program is shown in Figure 26.

The following numbers correspond to the numbers on Figure 26. A program cycle begins with item 1 and continues through item 11. Steps 6 and 7 are referred to as total time. Steps 11 and 1 are referred to as detail time.

1. Before the first record is read, the object program prepares and prints any heading information to be printed on the first page. After the first record has been read, the object program prepares and prints any heading and detail information which is not conditioned on overflow.

2. The object program tests for any halt indicators. If any halt indicators are on, the program branches to item 12.

3. The object program tests for the end-of-file conditions. If the end-of-file condition has occurred, the program branches to item 13.

4. The object program then reads an input record.

5. Starting with line 1 of the Input Specification sheet, and with the record just read, the object program uses the record identification code to identify the record. When the identification code matches an entry on the input sheet, the object program turns on the resulting indicator that has been specified for the record. When a control-field break occurs, appropriate control-level indicators are turned on.

6. Next, all total calculations are performed. (This step is bypassed for the initial control break which is caused by reading the first input record.)

7. Next, all total output lines which are not conditioned on overflow are prepared and printed. (This step is also bypassed for the first control break.)

8. The object program tests for the last record indicator. If it is on, the program branches to item 14.

9. The object program tests for an overflow condition. If an overflow condition has occurred, the program branches to item 15.

10. The data fields contained in the input record just read are moved into storage. These fields are specified by field entries on the Input Specification sheet.

11. Any detail calculations are performed, and processing continues with item 1.

12. Program execution stops.

13. The Last-Record indicator (LR) is set ON and all control-level indicators L1-L9 are set ON. Then the program branches to item 6.

14. Program execution stops.

15. If overflow has occurred, any lines conditioned by overflow are printed. If no overflow printing has been specified, an automatic skip to channel 1 occurs. The program then branches to item 10.

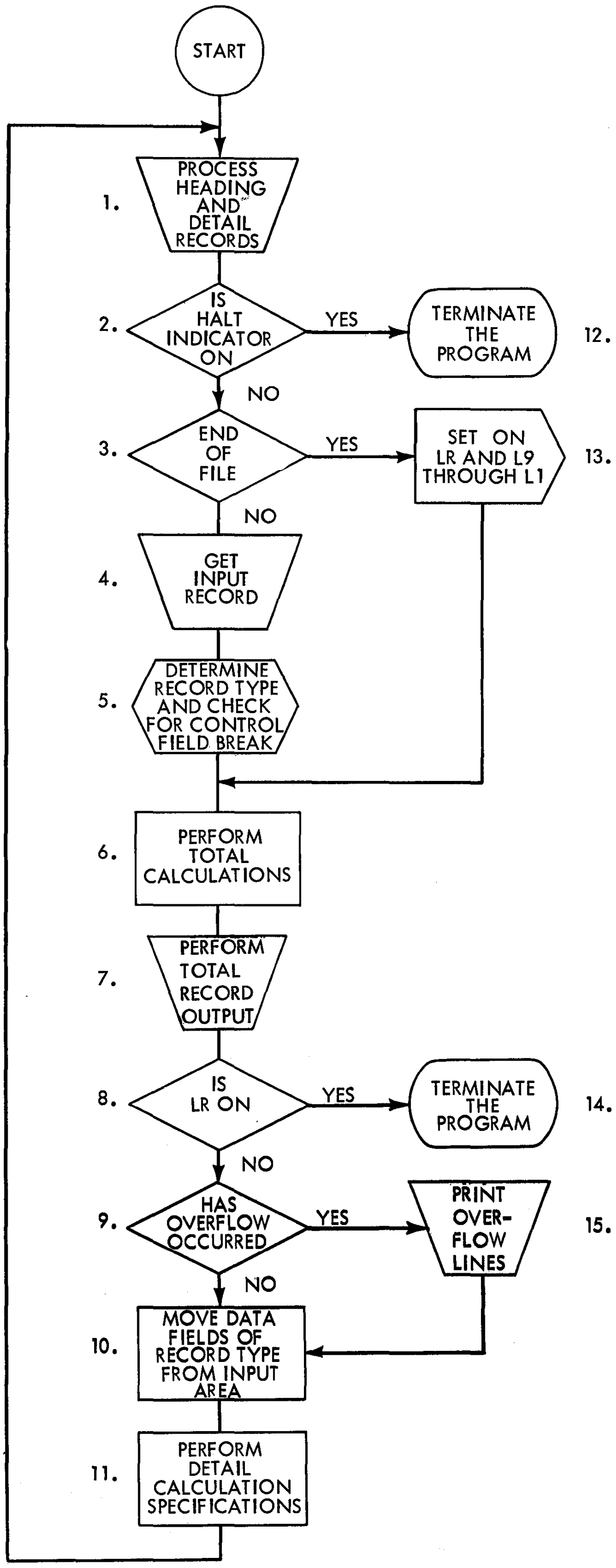

START
1.
PROCESS HEADING AND DETAIL RECORDS
2.
IS HALT INDICATOR ON
YES
TERMINATE THE PROGRAM
12.
NO
3.
END OF FILE
YES
SET ON LR AND L9 THROUGH L1
13.
NO
4.
GET INPUT RECORD
5.
DETERMINE RECORD TYPE AND CHECK FOR CONTROL FIELD BREAK
6.
PERFORM TOTAL CALCULATIONS
7.
PERFORM TOTAL RECORD OUTPUT
8.
IS LR ON
YES
TERMINATE THE PROGRAM
14.
NO
9.
HAS OVERFLOW OCCURRED
YES
PRINT OVER-FLOW LINES
15.
NO
10.
MOVE DATA FIELDS OF RECORD TYPE FROM INPUT AREA
11.
PERFORM DETAIL CALCULATION SPECIFICATIONS

Figure 26. General Logic Flow of an RPG Object Program

PROBLEM DEFINITION

The programming examples in the preceding section were intended to introduce the reader to the use of RPG and were therefore kept simple. More complex applications may require a thorough analysis of the existing or proposed system before program writing.

This analysis should include a description of source data and its format, and how this data should be processed to develop the report and other necessary output information.

The following types of information must be defined before coding the program:

1. The available data.

2. The input and output formats to be used.

3. The information required in the input and output formats.

4. The codes to be used to identify the various inputs and outputs and their elements.

5. The handling of the various transactions and exceptions.

After all application data has been gathered, document it for easy reference during the writing of the specification forms. One method of documenting an application is to lay out the complete format of the report on a printer spacing chart. This method also provides a pictorial representation of the final product.

PRINTER SPACING CHART

Before the report specifications are written, the programmer should have a clear picture of what he wants as the final product. If the report is to be printed, he must know the number of fields to be placed on each line of the report, the spacing between lines, and the positioning of the information within each line of the report.

Although no cards for the source deck are punched directly from the entries on this chart, the representation serves as a guide for completing the specification sheets. It plays an important role in writing report specifications. If the final product is written on magnetic tape or direct-access storage devices or if it is punched in cards, the user must know where the information is to be located. A tape layout chart (Form X21-9011) or a direct-access storage-device layout chart (Form X24-1711) can be used.

Layout of Lines and Fields

The two most important functions of a printer spacing chart are:

1. To establish the positions of the data to be printed and to indicate the spacing between printed lines.

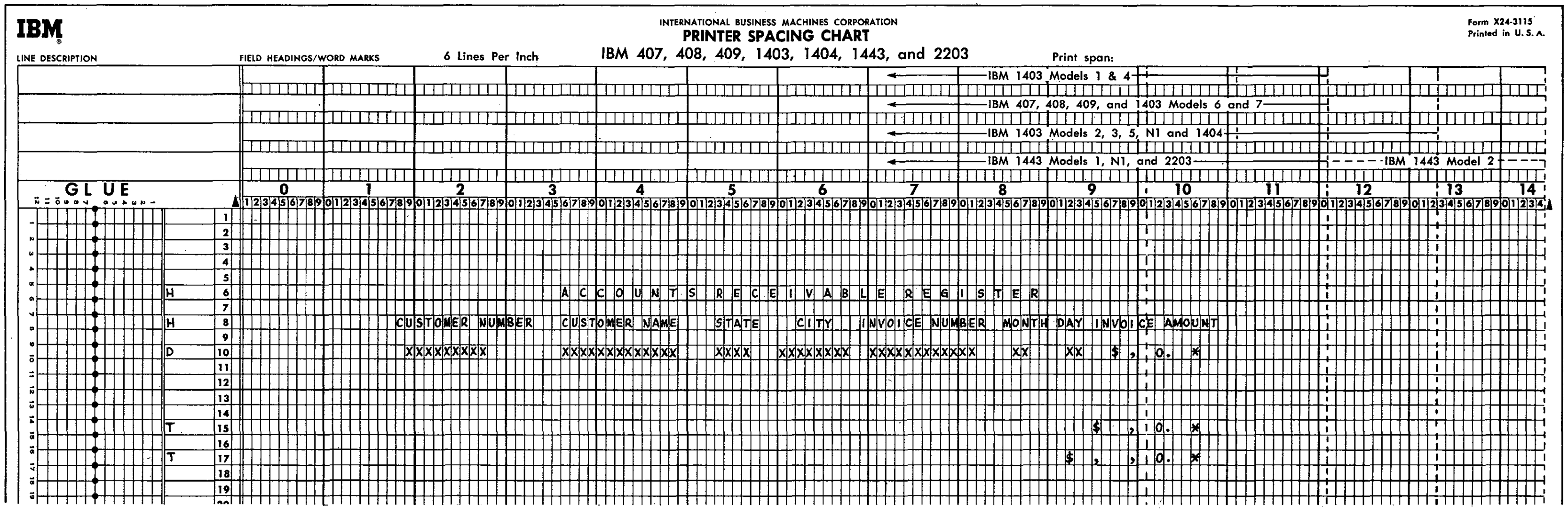
2. To assign each line an identification code representing the type of line. Figure 27 shows an example of the printer spacing chart (Form X24-6436).

The numbers across the top and bottom of the spacing chart represent the print positions. The numbers down the left side are the line numbers. The programmer selects the line number and print positions for a particular field and makes his notation in the selected positions.

Headings and other constant information are spelled out completely in the print positions assigned to them. Variable information is represented by X's and where applicable, includes credit symbols, punctuation, etc. The position in an amount field where zero suppression ends is indicated by a zero rather than an X.

Line-Identification Code

The line-identification code specifies the type of line to be printed. The identification codes are: H for a heading line, D for a detail line, and T for a total line. All lines must be identified as belonging to one of these categories.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
PRINTER SPACING CHART
IBM 407, 408, 409, 1403, 1404, 1443, and 2203

Form X24-3115
Printed in U. S. A.

LINE DESCRIPTION

FIELD HEADINGS/WORD MARKS

6 Lines Per Inch

Print span:
IBM 1403 Models 1 & 4
IBM 407, 408, 409, and 1403 Models 6 and 7
IBM 1403 Models 2, 3, 5, N1 and 1404
IBM 1443 Models 1, N1, and 2203
IBM 1443 Model 2

GLUE

```
 6  H                                    A C C O U N T S   R E C E I V A B L E   R E G I S T E R
 8  H                  CUSTOMER NUMBER   CUSTOMER NAME    STATE    CITY   INVOICE NUMBER  MONTH DAY INVOICE AMOUNT
10  D                   XXXXXXXXX        XXXXXXXXXXXXX    XXXX   XXXXXXXX  XXXXXXXXXXXX    XX    XX    $ ,  0.  *
15  T                                                                                                  $   ,  0.  *
17  T                                                                                               $  ,   ,  0.  *
```

Figure 27. Printer Spacing Chart

RPG SPECIFICATIONS SHEETS

This section provides detailed explanations of each specification field in the five RPG coding forms.

CROSS-REFERENCES

To make this reference manual a more effective learning tool, numerous crossreferences are placed for readers not familiar with disk-storage processing and related functions.

Disk storage, table lookup, matching field, and chaining field operations and related functions are described apart from the detailed descriptions of specifications for them.

These general introductory descriptions, contained at the back of the manual, are intended for the reader as he encounters the related specifications for them throughout the manual. To facilitate locating them in the manual, all cross-references used are listed in the Index.

The section Disk Storage Concepts provides a general introduction to disk-file organization and processing, including terminology associated with these functions. Readers not familiar with these concepts may wish to review that section before beginning with RPG Specification Sheets.

GENERAL INFORMATION

The forms are listed in the sequence in which they are discussed in this publication.

Input Specifications
Calculation Specifications
Output-Format Specifications
File Description Specifications
File Extension Specifications

Figure 28 shows the five specifications sheets.

INPUT SPECIFICATIONS: This form is used to:

1. Specify the file or files to be read into the system.

2. Identify the different types of records contained in each file.

3. Describe the location of the data fields in each record.

CALCULATION SPECIFICATIONS: This form specifies the operations to be performed upon the input data and upon data obtained as the result of previous calculations.

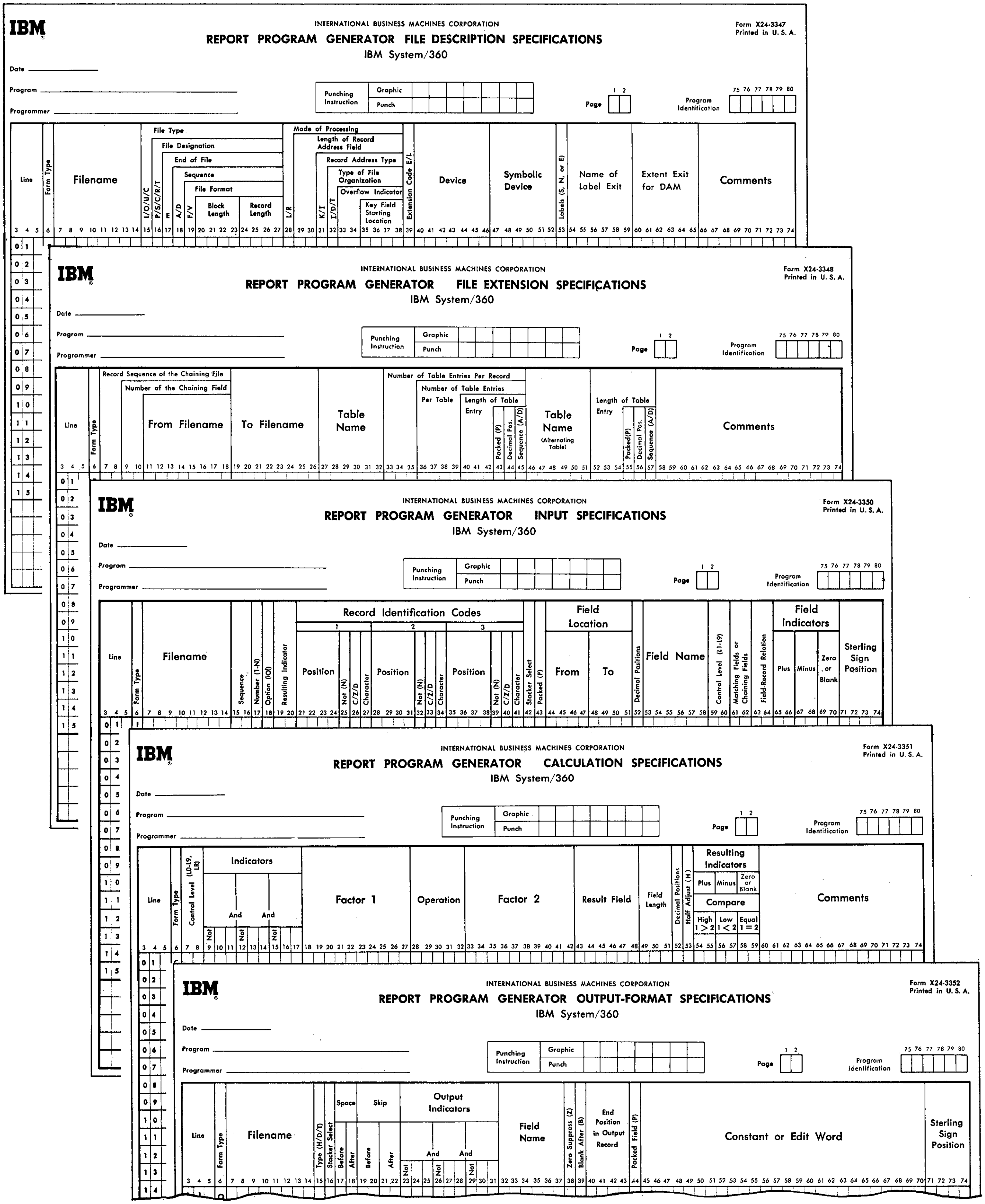
OUTPUT-FORMAT SPECIFICATIONS: This form specifies:

1. The kind of output files to be produced: printed reports, summary records, etc.

2. The location of the data fields in the output reports and records.

FILE DESCRIPTION SPECIFICATIONS: This form provides additional information about input and output files that is not included on the input or output sheets.

FILE EXTENSION SPECIFICATIONS: This form provides additional information about files used by the object program, such as tables, chaining files, and record-address files.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic, Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	File Type I/O/U/C	File Designation P/S/C/R/T	End of File E	Sequence A/D	File Format F/V	Block Length	Record Length	L/R	Mode of Processing	Length of Record Address Field	Record Address Type K/I	Type of File Organization I/D/T	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	Comments
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

Lines: 0 1, 0 2, 0 3, 0 4, 0 5, 0 6, 0 7, 0 8, 0 9, 1 0, 1 1, 1 2, 1 3, 1 4, 1 5

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Form X24-3348
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic, Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Record Sequence of the Chaining File	Number of the Chaining Field	From Filename	To Filename	Table Name	Number of Table Entries Per Record	Number of Table Entries Per Table	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Table Name (Alternating Table)	Length of Table Entry	Packed(P)	Decimal Pos.	Sequence (A/D)	Comments
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

Lines: 0 1, 0 2, 0 3, 0 4, 0 5, 0 6, 0 7, 0 8, 0 9, 1 0, 1 1, 1 2, 1 3, 1 4, 1 5

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic, Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Record Identification Codes: 1, 2, 3

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators Plus	Field Indicators Minus	Field Indicators Zero or Blank	Sterling Sign Position
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

Lines: 0 1, 0 2, 0 3, 0 4, 0 5, 0 6, 0 7, 0 8, 0 9, 1 0, 1 1, 1 2, 1 3, 1 4, 1 5

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic, Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Control Level (L0-L9, LR)	Indicators Not		And Not		And Not		Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators Plus / Compare High 1 > 2	Minus / Low 1 < 2	Zero or Blank / Equal 1 = 2	Comments
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

Lines: 0 1, 0 2, 0 3, 0 4, 0 5, 0 6, 0 7, 0 8, 0 9, 1 0, 1 1, 1 2, 1 3, 1 4, 1 5

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic, Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators Not		And Not		And Not		Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
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

Lines: 0 1, 0 2, 0 3, 0 4, 0 5, 0 6, 0 7, 0 8, 0 9, 1 0, 1 1, 1 2, 1 3, 1 4

Figure 28. RPG Specification Sheets

COMMON FIELDS

There are five entries that have the same function in all five forms. These are described first.

PAGE NUMBER (COLUMNS 1-2)

Each specification page of the source program must be numbered. The pages are numbered beginning with 01 for the file-description form, and continuing in the following sequence:

File Extension

Input

Calculation

Output-Format

LINE NUMBER (COLUMNS 3-5)

Each specification line must be identified by a line number. The first two digits of the line number are preprinted on the form. The third position (column 5) is used when, after all specifications have been written on the form, it becomes necessary to insert an additional line between two previously written lines. The line to be inserted is written following line 15. It is given an appropriate number (and subnumber in column 5).

The cards must be in numeric sequence by columns 1-5 when they are read by RPG. This means that the specification cards for a program could be placed in numeric sequence (if, for example they were accidentally dropped or upset) by sorting or arranging them in sequence by page number and line number.

FORM TYPE (COLUMN 6)

Each form has an appropriate type-code preprinted in column 6. This code must be punched into all specifications cards. The codes are:

I - Input

C - Calculation

O - Output-Format

F - File Description

E - File Extension

COMMENTS (COLUMN 7)

This feature enables the programmer to insert an identifying comment in the specification sheets. This facility can be used, for example, to identify the end of one section of a program. These comments are written on the specification line, preceded by an asterisk in column 7. During the generation of the object program, the asterisk in column 7 identifies the comment so that it is not considered a specification.

PROGRAM IDENTIFICATION (COLUMNS 75-80)

This entry identifies the specification cards for a particular program or for a specific section of a large program.

GENERAL INFORMATION

The specifications for this sheet are divided into two categories as shown in Figure 29.

1. Record Identification (columns 7-42). These entries identify the input record (by specifying the identifying record codes it contains) and specify the relationship of the record to other records in the file.

2. Field Description (columns 43-74). These entries describe the fields of the input record used in the report. Each field is described on a separate line and is written on the line below its corresponding record-identification entry.

RECORD IDENTIFICATION ENTRIES

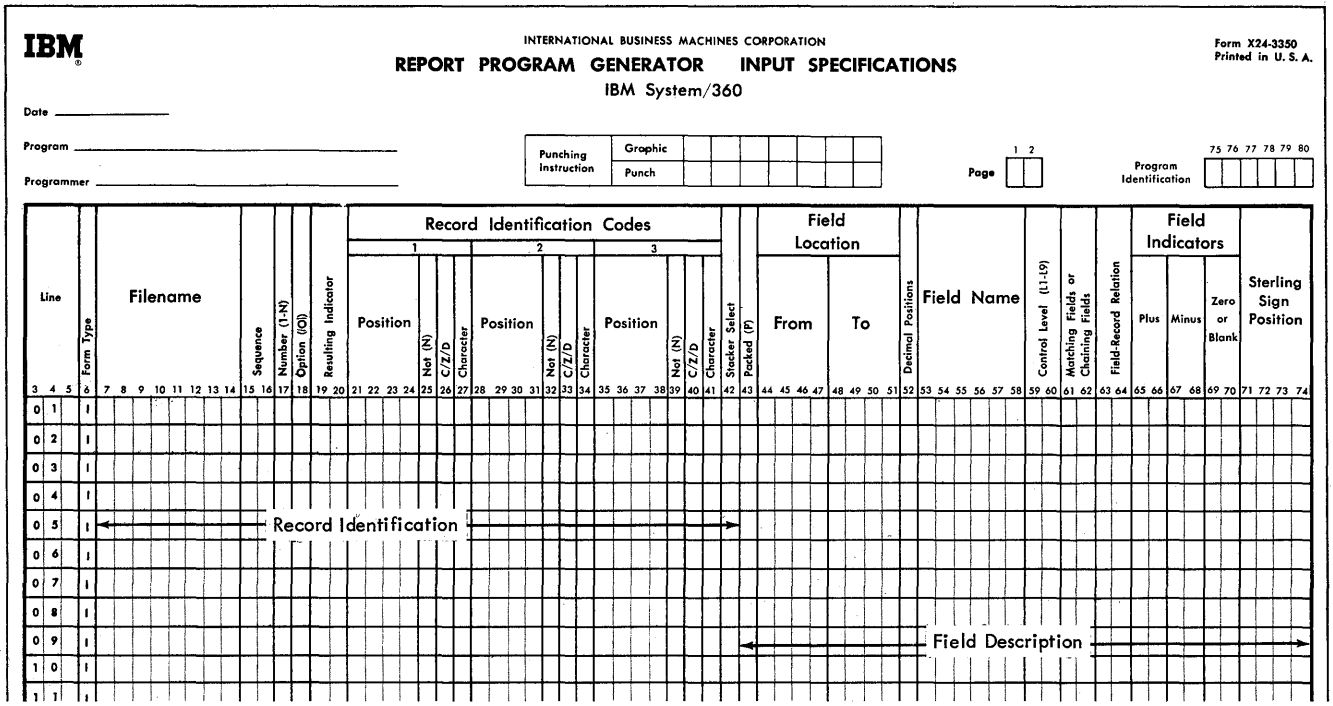
FILE NAME (COLUMNS 7-14)

A file name must be given to each input file. The file name must be left-justified (that is, it must start in column 7) and it must begin with an alphabetic character. The file name may be alphameric, but it must not contain special characters or blanks. The file name may be eight characters or fewer.

> Note 1: In this publication, an alphabetic character refers to the letters A-Z, and the dollar sign, pound sign, and the at sign ($, #, and @).

The file name must be entered only with the first record-identification line of the appropriate file.

> Note 2: The file name may be eight characters long but RPG will utilize only the first seven characters for all files except indexed-sequential files. For indexed-sequential files, only the first five characters of the filename



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3 4 5) | Form Type (6) | Filename (7 8 9 10 11 12 13 14) | Sequence (15 16) | Number (1-N) (17) | Option (O) (18) | Resulting Indicator (19 20) | Record Identification Codes: 1 — Position (21 22 23 24), Not (N) (25), C/Z/D (26), Character (27); 2 — Position (28 29 30 31), Not (N) (32), C/Z/D (33), Character (34); 3 — Position (35 36 37 38), Not (N) (39), C/Z/D (40), Character (41) | Stacker Select (42) | Packed (P) (43) | Field Location: From (44 45 46 47), To (48 49 50 51) | Decimal Positions (52) | Field Name (53 54 55 56 57 58) | Control Level (L1-L9) (59 60) | Matching Fields or Chaining Fields (61 62) | Field-Record Relation (63 64) | Field Indicators: Plus (65 66), Minus (67 68), Zero or Blank (69 70) | Sterling Sign Position (71 72 73 74)

0 1 I
0 2 I
0 3 I
0 4 I
0 5 I ← Record Identification →
0 6 I
0 7 I
0 8 I
0 9 I ← Field Description →
1 0 I
1 1 I

Figure 29. Input Specification Sheet

are utilized in the RPG program. Therefore, it is important that the first seven (or five) characters provide a unique filename.

SEQUENCE (COLUMNS 15-16)

This entry designates the sequence of input record-types.

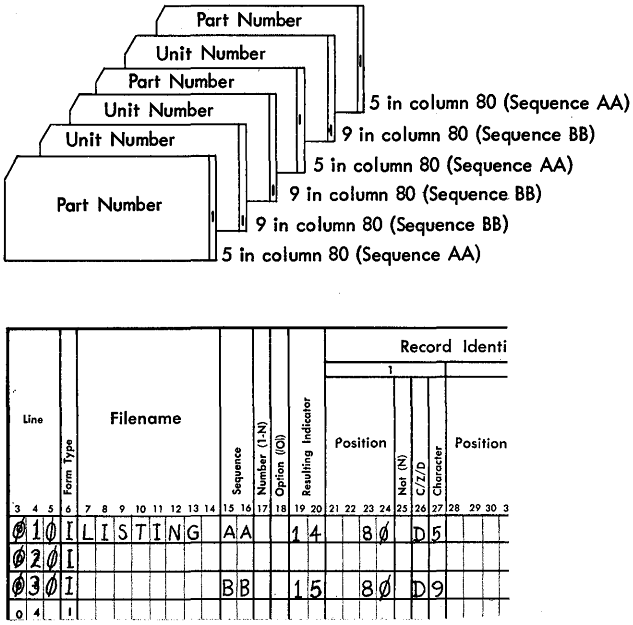
Alphabetic Entries

If the input records are not required to be in a predetermined sequence within a control group, or if it is not necessary to terminate processing when the input records are not in sequence, the entry in columns 15-16 should be a two-character alphabetic code.

In Figure 30, the input data records, which contain part numbers and unit number, do not appear in a predetermined sequence with in a control group. Consequently, each input record-type is assigned an alphabetic code because the order of processing is not important to the program.

Numeric Entries

If there is more than one record type within an input file, the records may appear in some predetermined sequence. To process the input data records in a predetermined sequence, the input records are specified on the Input Specification sheet in the same sequence in which they are to be read by the object program. For example, assume that the input records appear as shown in Figure 31. The Name Record is assigned 01 because it is the first type. The Street Record is the second type. The Street Record is assigned 02. The City-State Record is assigned 03, and the Item Number Record is assigned 04. (In this case, there may be several items-number records). If the input records do not appear in this predetermined sequence, program processing stops (information is out of sequence).


Part Number
Unit Number
Part Number
Unit Number
Unit Number
Part Number
5 in column 80 (Sequence AA)
9 in column 80 (Sequence BB)
5 in column 80 (Sequence AA)
9 in column 80 (Sequence BB)
9 in column 80 (Sequence BB)
5 in column 80 (Sequence AA)

Line 3 4 5	Form Type 6	Filename 7 8 9 10 11 12 13 14	Sequence 15 16	Number (1-N) 17	Option (O) 18	Resulting Indicator 19 20	Record Identi 1 Position 21 22 23 24	Not (N) 25	C/Z/D 26	Character 27	Position 28 29 30 3
Ø1Ø	I	LISTING	AA			14	8Ø		D	5	
Ø2Ø	I										
Ø3Ø	I		BB			15	8Ø		D	9	
04											

Figure 30. Input Data Records

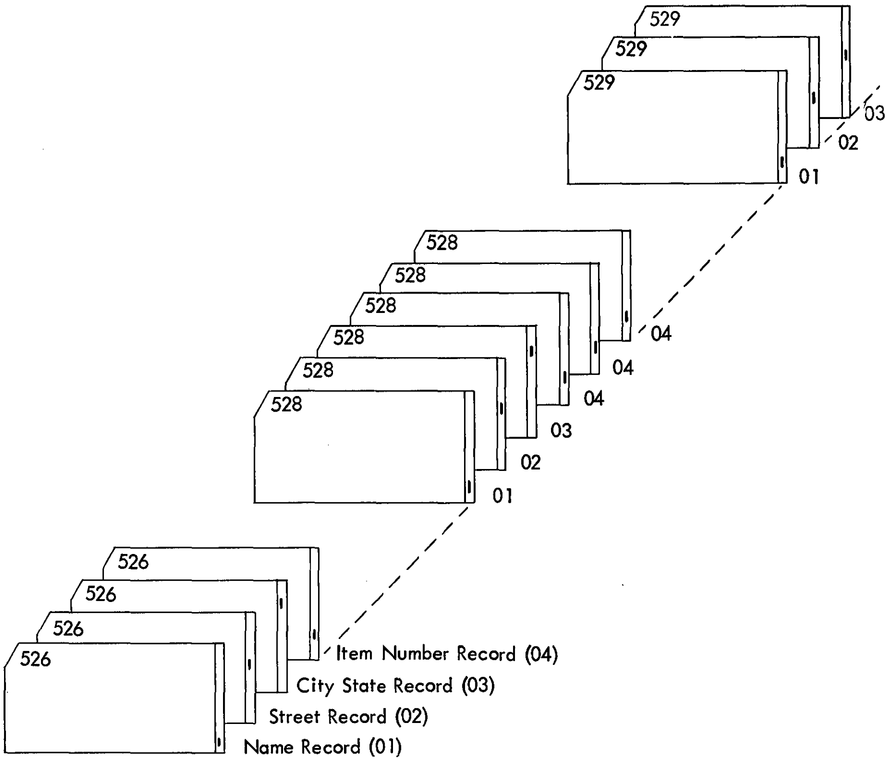
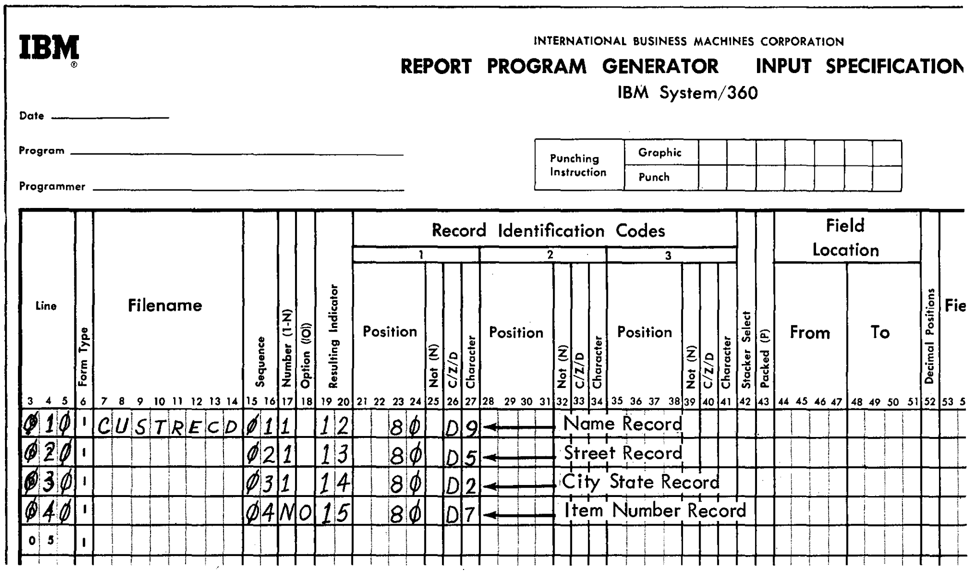


Figure 31. Input Records in Sequence



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATION

IBM System/360

Date

Program

Programmer

Punching Instruction	Graphic							
	Punch							

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1 Position	Not (N)	C/Z/D	Character	2 Position	Not (N)	C/Z/D	Character	3 Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	Field Location From	To	Decimal Positions	Fie
Ø1Ø	I	CUSTRECD	Ø1	1		12	8Ø		D	9	← Name Record													
Ø2Ø	I		Ø2	1		13	8Ø		D	5	← Street Record													
Ø3Ø	I		Ø3	1		14	8Ø		D	2	← City State Record													
Ø4Ø	I		Ø4	N	O	15	8Ø		D	7	← Item Number Record													
0 5	I																							

Figure 32. Entries for Numeric Sequence

Figure 32 illustrates the sequence in which the records must appear.

Numeric entries must consist of two digits.

Restrictions

- The entries in a predetermined sequence must begin with 01 for each file (columns 15-16 of the input sheet).

- The entries in a predetermined sequence must be in ascending order in columns 15-16.

- Header cards or other cards that are not in sequence must be specified on the Input Specification sheet before specifications that are to be sequence-checked. Enter alphabetic sequence record-types first, followed by numeric-sequence record-types for each file.

- A maximum of 99 record-types is allowed.

NUMBER (COLUMN 17)

If a numeric code is assigned under Sequence, an entry must also be made in Number (column 17). If an alphabetic code has been assigned under Sequence, leave this column blank.

The entries for Number (column 17) are:

1 - Only one record of a type may be present in a control group.

N - One or more records of a type may be present in a control group.

Blank - The records do not appear in a determined sequence.

As shown in Figure 32, there can only be one Name Record, but there can be multiple Item Number Records.

OPTION (COLUMN 18)

This specification is used only with numeric-sequenced record-types. If a specific record-type must be present to perform an operation, leave this column blank. If the presence of a record is optional, enter the letter O in column 18.

In Figure 32, the O (alphabetic) in column 18 indicates there may or may not be an Item Number for each group. In this example, Name, Street, and City-State records must be present.

RESULTING INDICATOR (COLUMNS 19-20)

The entry in these columns is used in conjunction with the next entry Record Identification Codes. This entry serves two functions:

1. It establishes a unique two-digit code for each input-record type.

2. A special indicator is turned on in the object program each time the input record is present. That is, the object program may test for this condition during the processing of the calculation and output specifications.

To understand the first function of this entry, assume that an input record is identified by all of the following codes:

1. An X in column 80.
2. A digit 5 in column 79.
3. No R in column 78.

By assigning one two-digit resulting-indicator code to represent all these codes, it is easier to refer to this input record during the processing of the calculation and output specifications.

As an example of the second function of this entry, resulting indicators may be compared to selectors in punched-card machines or to internal and external switches on electronic data processing machines. The use of resulting indicators (like the use of selectors and switches) permits certain operations to occur only on specific conditions.

In Figure 32, the four record-types have been assigned resulting indicators 12, 13, 14, and 15. When one of the records is present, the appropriate resulting indicator is turned on, and those specifications pertaining to the record are performed. Specifications associated with other record-types are not performed.

Resulting indicators are turned on and off during the processing of the object program, as the various record-types are read by the system. However, only one resulting indicator (for a record-type) can be on at one time. When a resulting indicator is turned on, all other resulting indicators are turned off. There is one exception to this condition (see Chaining).

Other indicator conditions that can be established in the program are field indicators (columns 65-70 of the Input Specification sheet) and resulting indicators (columns 54-59 of the Calculation Specification sheet). These functions are described later, but one aspect of their use is of interest at this time.

All three types of indicators are assigned a two-digit number from 01-99. The indicator codes can be assigned in any sequence. For example, four different record-types could be assigned codes 40, 62, 98, and 02.

RECORD-IDENTIFICATION CODES (COLUMNS 21-41)

These entries provide a way of identifying each different record-type used in the object program. As mentioned previously, once the record-type has been defined on the Input Specification sheet, references to the record are made by its resulting indicator.

These columns provide for the entry of one to three identifying codes as indicated by the numbers 1, 2, and 3 on the Input Specification sheet. It is possible to specify more than three codes by using more than one line. If only one record-type is in the input file, the identification codes can be left blank, but a resulting indicator and sequence number must still be specified.

Each of the three sets of entries is the same, so only the first set (columns 21-27) is described. Each set is divided into four categories:

Position (columns 21-24)
Not (column 25)
C/Z/D (column 26)
Character (column 27)

Position (Columns 21-24)

Enter in this column (or columns) the position in each data record of the character that contains the identifying code. The position must be right-justified in columns 21-24, and leading zeros may be omitted.

Not (Column 25)

Enter an N in this column if the code described must not be present in the specified record position. Otherwise, leave this column blank.

C/Z/D (Column 26)

The object program identifies the different record-types of a program by comparing the character written in Character against the codes contained in the records. The entry in column 26 specifies whether the object program is to compare against the entire character (C), against only the zone portion (Z), or against only the digit (D). Enter a C, Z or D in column 26.

ZONE RECORD IDENTIFICATION: Testing a zone means that the zone of the character specified in columns 21-24 of the input sheet is used to identify the record. The ampersand, the minus, and the blank are exceptions. An ampersand will be identified as a 12-zone, the minus sign as an 11-zone, and a blank will be identified as a no-zone. The four common zones are:

+

1. 12-zone or plus zone (0, A-I, and &).

-

2. 11-zone or minus zone (0, J-R, and minus).
3. 0-zone (S-Z).
4. No zone (0-9 and Blank).

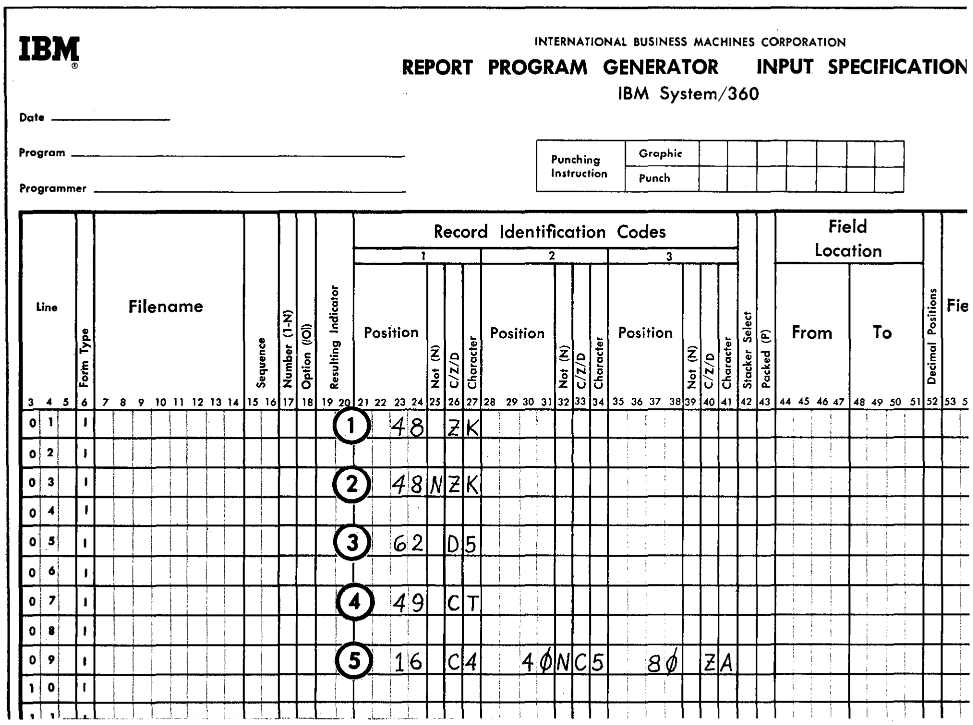
When you use a blank, ampersand, or minus in the character portion of a zone test, other characters which contain the same machine zone (e.g. ?, $, or %, respectively) will not satisfy the zone test.

Character (Column 27)

Enter in this column the identifying character that will be compared to the character specified in the input record. The character used in this column may be any letter A-Z, any number 0-9, or any special character.

Examples of Record-Identification Codes

Examples of record-identification specifications are shown in Figure 33. The explanation of each entry is given here.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATION

IBM System/360

Date

Program

Programmer

Punching Instruction	Graphic							
	Punch							

							Record Identification Codes														Field Location			
							1				2				3									
Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	From	To	Decimal Positions	Fie
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 5
0 1	I					1	48		Z	K														
0 2	I																							
0 3	I					2	48	N	Z	K														
0 4	I																							
0 5	I					3	62		D	5														
0 6	I																							
0 7	I					4	49		C	T														
0 8	I																							
0 9	I					5	16		C	4	4Ø	N	C	5	8Ø		Z	A						
1 0	I																							

Figure 33. Record Identification Codes

1. This entry specifies an 11-zone in record position 48. Any of the letters J-R, or the minus sign, could be placed in Character because they all contain an 11-zone. A Z must be placed in column 26 so that only the zone portion is checked. If a C were placed in this column, the object program would compare the letter K against the input-record code instead of a 11-zone. A minus could be placed in this field to represent an 11-zone, and an ampersand could represent a 12-zone.

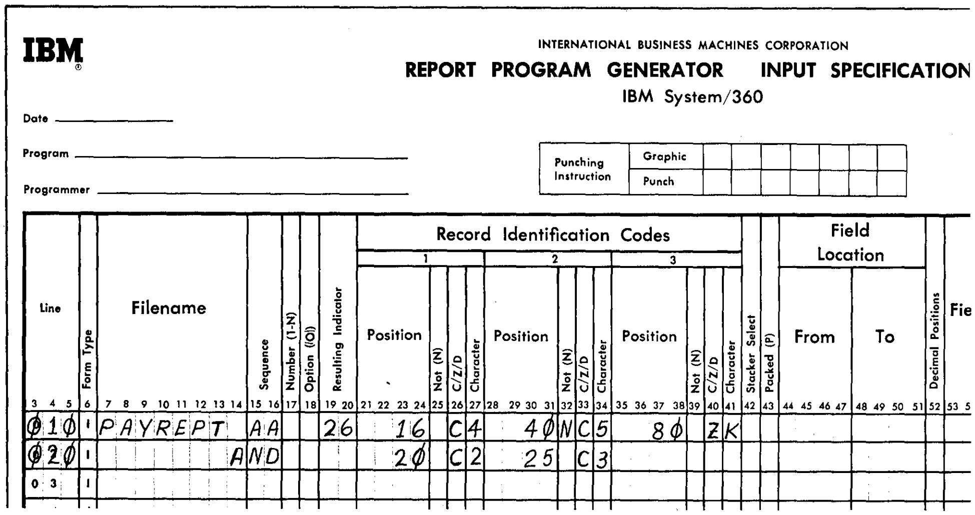
2. This entry specifies that no 11-zone may be present in record-position 48.

3. This entry specifies that the digit portion of the code is checked. A character whose digit portion is 5 must be in record-position 62. For example, a 5, N, V, or E would fulfill this requirement.

4. This entry specifies that the letter T must be present in record-position 49.

5. These entries specify that all three codes must appear in the same record. A 4 must appear in column 16. Column 40 must not contain the number 5, and column 80 must contain a 12-zone. This is known as an AND relationship.

The last example illustrates the way to specify more than one code on more than one line. If an input record contained five different codes, they could be specified as shown in Figure 34. This is also known as an AND relationship. It implies that the card is identified by:



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATION

IBM System/360

Date

Program

Programmer

Punching Instruction	Graphic							
	Punch							

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes												Stacker Select	Packed (P)	Field Location		Decimal Positions	Fie
							1				2				3									
							Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character			From	To		
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 5
Ø1Ø	I	PAYREPT	AA			26	16		C	4	4Ø	N	C	5	8Ø		Z	K						
Ø2Ø	I	A	ND				2Ø		C	2	25		C	3										
0 3	I																							

Figure 34. Specifying More Than Three Identification Codes

1. A 4 in column 16, and
2. No 5 in column 40, and
3. An 11-zone in column 80, and
4. A 2 in column 20, and
5. A 3 in column 25.

Additional specification lines can be used to specify as many record identification codes as required. Each additional line must begin with the word AND in columns 14-16 and blanks in columns 17-20.

The use of this facility and its advantages will be easier to understand after the explanation of Records in an OR-Relationship.

OR Relationship

Three types of OR relationships are possible:

1. One or more record-types have the same fields in the same positions of the record. (All fields in one record-type are in the same relative positions in another record-type.)

2. One or more record-types have the same fields, but the fields are in different positions of the record. For example: unit cost is in positions 21-25 in one record-type and in positions 31-35 in another record-type.

3. One or more record-types have different fields in the same positions or in different positions of the record. For example: two record-types with ten fields in the same relative positions, but with three fields in different positions. It is of value to specify these record-types in an OR relationship only if there are more fields that are alike than fields that are unlike.

Omitting Record Identification

When input records are to be processed alike without regard for their identifying codes, columns 21-41 may be left blank; however, sequence (columns 15-16) and a resulting indicator (columns 19-20) must be specified.

When only certain record-types within a group are to be read and processed, they must be listed first with their identifying record codes. The remaining record-types may be bypassed or processed as a group. In either case, they are specified as the last record-type and they must have a resulting indicator specified for them. Columns 21-41 of the Input Specification sheet are left blank.

> Note: If the records are to be bypassed, they should not be referred to on the calculation or output sheets.

STACKER SELECT (COLUMN 42)

If the input record type is from cards that are to be stacker-selected, the number of the stacker into which the card is to be selected is entered in column 42. Leave column 42 blank if the cards are to go to the normal pocket.

The stacker pockets and their acceptable codes are:

IBM 1442	
Pocket	Code
1	1, or blank
2	2

IBM 2501	
Pocket	Code
Leave blank	

IBM 2520	
Pocket	Code
1	1, or blank
2	2

IBM 2540	
Pocket	Code
Normal or R1	Blank
R2	1
RP3	2

The stacker-select entry should be made on the same line with the record identification. If different records in an OR relationship are to be stacker-selected, enter the codes on the same line with the record identification. Stacker select must be specified on each line in an OR-relationship if it is desired.

COMBINED FILES STACKER SELECT

When there are combined files to be stacker-selected, the user can determine the stacker pocket into which the cards will fall. The Stacker Select entries (column 42 of the Input Specification sheet and column 16 of the Output-Format Specification sheet) must be utilized in one of the following ways when using an IBM 1442 Card Read-Punch:

1. If stacker pocket 1 is selected, enter 1 in the appropriate column of either sheet. The remaining entry can be 1 or blank.

2. If stacker pocket 2 is selected, enter 2 in the appropriate column of either sheet. The remaining entry can be 1, 2, or blank.

   Note: When there is to be punched input and punched output, stacker pocket 2 always takes precedence; i.e., it can never be overridden.

When an IBM 2540 Card Read-Punch is used, and the combined files are to be stacker selected, the last entry specified always takes precedence.

FIELD DESCRIPTION ENTRIES

As mentioned previously, the Input Specification sheet consists of two categories: record identification and field description.

On the record identification portion of the Input Specification sheet, one line represents the specifications for one record-type. On the field-description portion of the sheet, one line represents the specifications for one field of a record.

The following information concerns the individual field descriptions of one record-type. Field descriptions are always written on the specification line immediately below the specification line that identifies the record-type.

Field description entries describe the fields of the input record to be used in the report. Each field of the record requires one line on the Input Specification sheet. Columns 7-42 of the line must be blank.

PACKED (COLUMN 43)

Enter a P in this column if the input field is in the packed decimal format. Otherwise, leave this column blank.

Note 1: If a P is entered in this column and the input field specified is blank, an arithmetic check and an abnormal end-of-job will occur.

Note 2: If the field is in packed format, RPG will compute the unpacked field length using the following formula:

[(TO Position-FROM Position)*2]+1

FIELD LOCATION

Columns 44-51 of the Input Specification sheet are used to describe the location of each field in the record. The maximum field length for a numeric field is 15 digits. The maximum field length for an alphameric field is 256 characters.

FROM (COLUMNS 44-47)

This entry shows the location of the first position of the field, that is, the high-order position. The entry in columns 44-47 is right-justified, and preceding zeros may be omitted.

TO (COLUMNS 48-51)

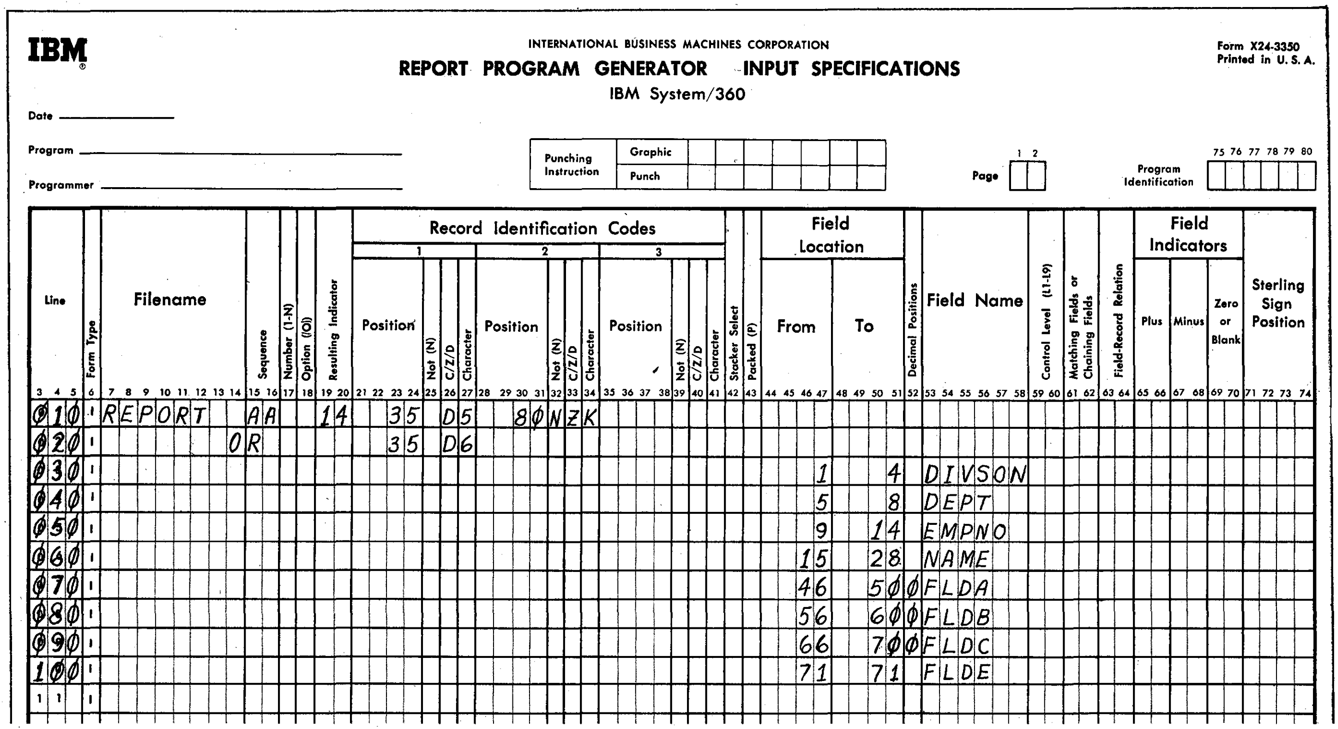
This entry shows the location of the last position of the field (the low-order position). The entry in columns 48-51 is right-justified, and preceding zeros may be omitted. In Figure 35, the field DEPT may be found in columns 5-8 of the input record.

RECORDS IN AN OR-RELATIONSHIP

It is possible to specify two different record-types (whose fields are in the same positions) with one set of specifications. In Figure 35 note that OR has been entered in columns 14-15. Thus, two different input records have the same fields, but the identifying codes are different. The OR in columns 14-15 indicates that the fields may be taken from an input record that has a 5 in column 35 and no 11-zone in column 80, or the field may be taken from an input record that has a 6 in column 35. Thus, Figure 35 shows the only additional specification required to specify both input records that are acceptable.

> Note: In this case, it is not necessary to enter indicator 14 for the second record identification, because it has been previously specified.

It is also possible to specify two records in an OR-relationship when the field records are not in the same location. Using Figure 36, notice in columns 19-20 that resulting indicator 14 is turned on when a 5 is in column 35 and no 11-zone is in column 80. Resulting indicator 16 is turned on when the input record has a 6 in column 35. By specifying an OR-relationship, it is possible to specify



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U.S.A.

Date

Program

Programmer

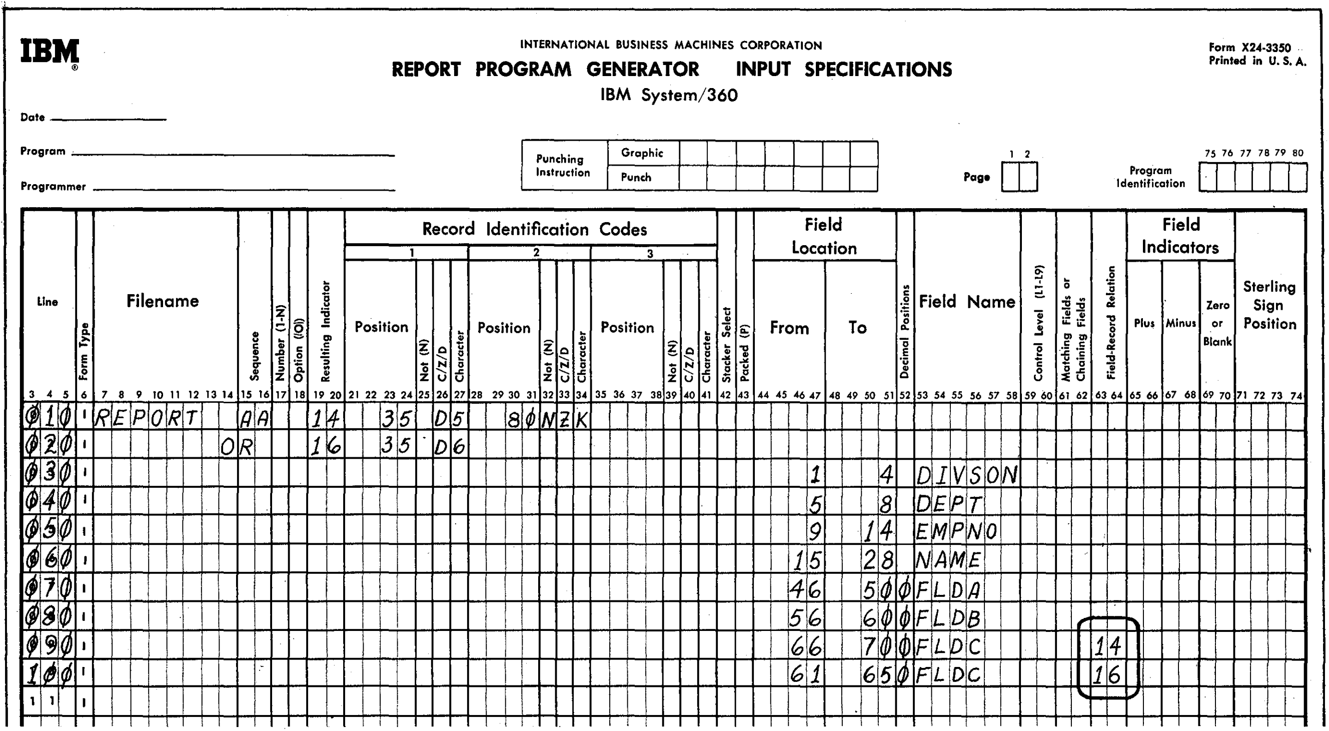
Punching Instruction	Graphic							
	Punch							

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1 Position	Not (N)	C/Z/D	Character	2 Position	Not (N)	C/Z/D	Character	3 Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	Field Location From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators Plus	Minus	Zero or Blank	Sterling Sign Position
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
Ø1Ø	I	REPORT	AA			14	35		D	5	8Ø	N	Z	K																	
Ø2Ø	I	O	R				35		D	6																					
Ø3Ø	I																				1	4		DIVSON							
Ø4Ø	I																				5	8		DEPT							
Ø5Ø	I																				9	14		EMPNO							
Ø6Ø	I																				15	28		NAME							
Ø7Ø	I																				46	5Ø	Ø	FLDA							
Ø8Ø	I																				56	6Ø	Ø	FLDB							
Ø9Ø	I																				66	7Ø	Ø	FLDC							
1ØØ	I																				71	71		FLDE							
1 1	I																														

Figure 35. Specifying Input Fields



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date ______

Program ______

Programmer ______

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

							Record Identification Codes 1				2				3						Field Location							Field Indicators			
Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Plus	Minus	Zero or Blank	Sterling Sign Position
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
ϕ1ϕ	I	REPORT	AA			14	35		D	5	8ϕ	N	Z	K																	
ϕ2ϕ	I	O	R			16	35		D	6																					
ϕ3ϕ	I																				1	4		DIVSON							
ϕ4ϕ	I																				5	8		DEPT							
ϕ5ϕ	I																				9	14		EMPNO							
ϕ6ϕ	I																				15	28		NAME							
ϕ7ϕ	I																				46	5ϕ	ϕ	FLDA							
ϕ8ϕ	I																				56	6ϕ	ϕ	FLDB							
ϕ9ϕ	I																				66	7ϕ	ϕ	FLDC			14				
1ϕϕ	I																				61	65	ϕ	FLDC			16				
1 1	I																														

● Figure 36. Specifying Field Location

both record-types with one set of specifications.

In this example, the field locations are not the same in both types, so it is necessary to provide a separate resulting indicator code for the second record type. The second line, columns 14-27, of Figure 36 indicates that there is a second input record that is acceptable. However, FLDC (field C) is not located in the same positions in both record-types, so it must be specified twice. Each of the two entries for FLDC must be related to the appropriate record type. This is accomplished by entering the resulting indicator code, which is taken from columns (63-64).

Thus, if resulting indicator 14 is on, FLDC will be taken from positions 66-70 of the input record. If resulting indicator 16 is on, FLDC will be taken from positions 61-65 of the input record.

Specifying record-types in an OR-relation is not limited to only two records. As many records as are required can be specified as having an OR-relationship.

DECIMAL POSITION (COLUMN 52)

An entry in this column serves two functions:

1. It is used by the object program to determine the number of decimal positions contained in the numeric field which is specified in Field Location.

2. It causes the field specified in field location to have 0-, 11-, or 12-zone bits removed from all positions of the field except the units position.

Enter in this column the digit that represents the number of decimal positions in the input data field. If the data is a whole number, enter 0 in this column. An entry in this column specifies this field is a numeric field. If the field contains alphameric data that is to be treated as alphameric data, leave this column blank.

In Figure 36, DIVSON is an alphameric field and no entry is made in column 52. However, FLDA through FLDC are numeric fields but they contain no decimals. A zero is entered in column 52 for these fields.

FIELD NAME (COLUMNS 53-58)

Each field defined must be given a field name. Once a name has been assigned to a field, other references to it are made by using the field name, rather than by using the specific record position each time. Thus, the record positions of the input fields are not needed when writing the calculation and output specifications.

The field name must begin with an alphabetic character, and it must start in column 53. The field name may be alphameric.

Two input files may have fields with the same field names. A field name is assigned only once by RPG. Two input files will use the same storage location for fields with identical names. This procedure is permissible when using RPG, but the programmer should be aware that possible errors in calculations or output may occur when two input files have the same field names. This is especially important when chaining fields are specified on the Input Specification sheet.

Certain restrictions apply to the selection of field names. The following names can only be used in certain instances, and cannot be used at any time other than that specified.

1. T A B _ _ _. These three characters can only be used as a prefix to the name of a table.
2. P A G E _ _. This prefix can only be used in reference to page numbering.
3. A L T S E Q. This name cannot be used as a field name.
4. C O N T D _. This name can only be used in conjunction with the conversion routine.
5. I N n n _ _. This name can only be used as an RLABL referencing an indicator.

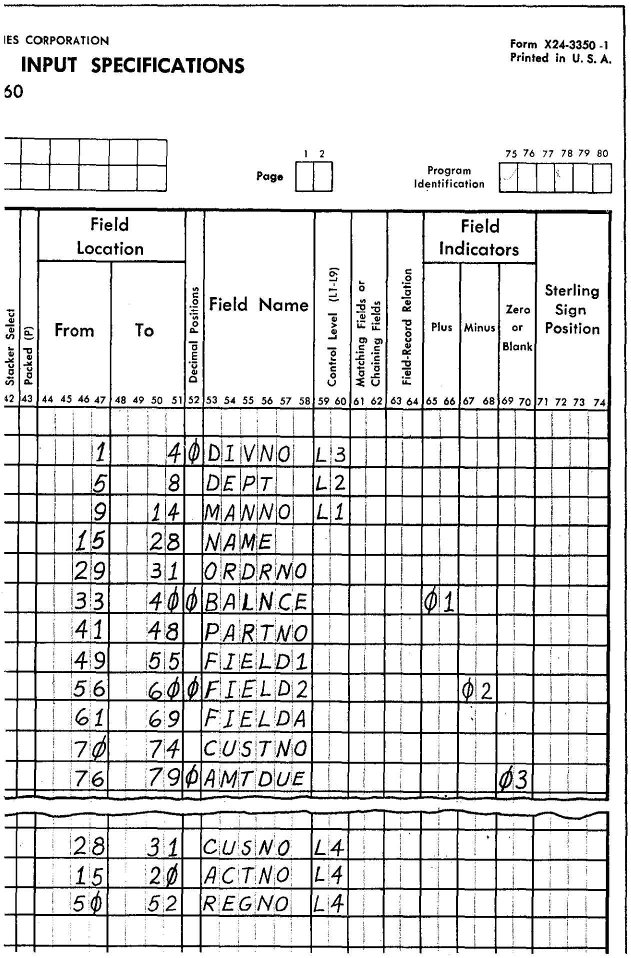
Specifying the Same Data Field as Alphameric and Numeric

If the same field from an input record is to be used as both an alphameric and a numeric field, the field must be specified twice by assigning two different field names to the same location in the record. If no decimal positions are specified for the field, it is considered to be an alphameric field.

Specifying Constants

The term constant is frequently used for information that is not changed with each record. It is usually not altered during the object run. Examples of constants are date cards, or other data that may be changed for each run. The technique to get this information to the program is by defining a special record type and by assigning a field name that is not otherwise used. If the constants must be available before the first data record of a file is processed, it may be necessary to define an additional file. This technique will also permit entering of constants that are too large to be specified as literals in the source program.

Figure 37 illustrates field names that are easy to read and suggest the function of the fields they represent.



IES CORPORATION

INPUT SPECIFICATIONS

Form X24-3350-1
Printed in U. S. A.

60

Page 1 2

Program Identification 75 76 77 78 79 80

Stacker Select	Packed (P)	Field Location		Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators			Sterling Sign Position
		From	To						Plus	Minus	Zero or Blank	
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
		1	4	Ø	DIVNO	L3						
		5	8		DEPT	L2						
		9	14		MANNO	L1						
		15	28		NAME							
		29	31		ORDRNO							
		33	4Ø	Ø	BALNCE				Ø1			
		41	48		PARTNO							
		49	55		FIELD1							
		56	6Ø	Ø	FIELD2					Ø2		
		61	69		FIELDA							
		7Ø	74		CUSTNO							
		76	79	Ø	AMTDUE						Ø3	
		28	31		CUSNO	L4						
		15	2Ø		ACTNO	L4						
		5Ø	52		REGNO	L4						

Figure 37. Field Names

CONTROL LEVEL (COLUMNS 59-60)

As the object program is processed, a change in a control field causes all processing indicated by this change to be initiated. Columns 59-60 are used to provide a convenient and simple method for specifying all control functions.

Up to nine control levels can be used by RPG. These levels are designated L1, L2, L3,...L9. An indicator similar to the resulting indicator is associated with each control-level designation. They are used to control functions specified on the calculation and output specification sheets. When the field specified in Field Location is a control field, its appropriate control level must be specified in columns 59-60. Figure 37 shows the entries for three control fields. (The control fields could have been listed on the sheet in any sequence.) When more than one file has control levels specified, the overall length of each control field must be the same length for all record types.

Additional Functions of Control Level Specifications

If a field specified in field location also has an entry in control level, the object program places the field into two main storage areas. One area, known as a control-field holding area, is used for the controlling functions of the field, and the other is used for any other uses of the field (such as for printing or for arithmetic calculations).

For example, it may be necessary to use a field for controlling functions but without considering zone bits in the comparison of the record to the next record. If the zone bits are not to be used in a comparison, specify the field as numeric by making a decimal entry in column 52. If the zone bits are to be used in a comparison, leave column 52 blank. In Figure 37, L3 in columns 59-60 causes the field DIVNO to be placed into storage in one area with zone-bits removed from all positions of the field to be used for control functions and to be placed in another area with zone bits removed from all positions of the field except the units position. The RPG program automatically performs the operations of storing the field twice and performing the appropriate zone elimination. No additional specifications need be written by the programmer.

SPLIT CONTROL FIELDS: Several fields in an input record can be specified as one control field. In the lower half of Figure 37, three fields, which are not in continuous record positions, are specified with the same L4 control level. The three fields are treated as one control field:

|CUSNO | ACTNO | REGNO|

The first field defined on the input sheet is placed in the high-order position, and the last field is placed in the low-order position. All fields are placed according to the sequence in which they are defined on the Input Specification sheet.

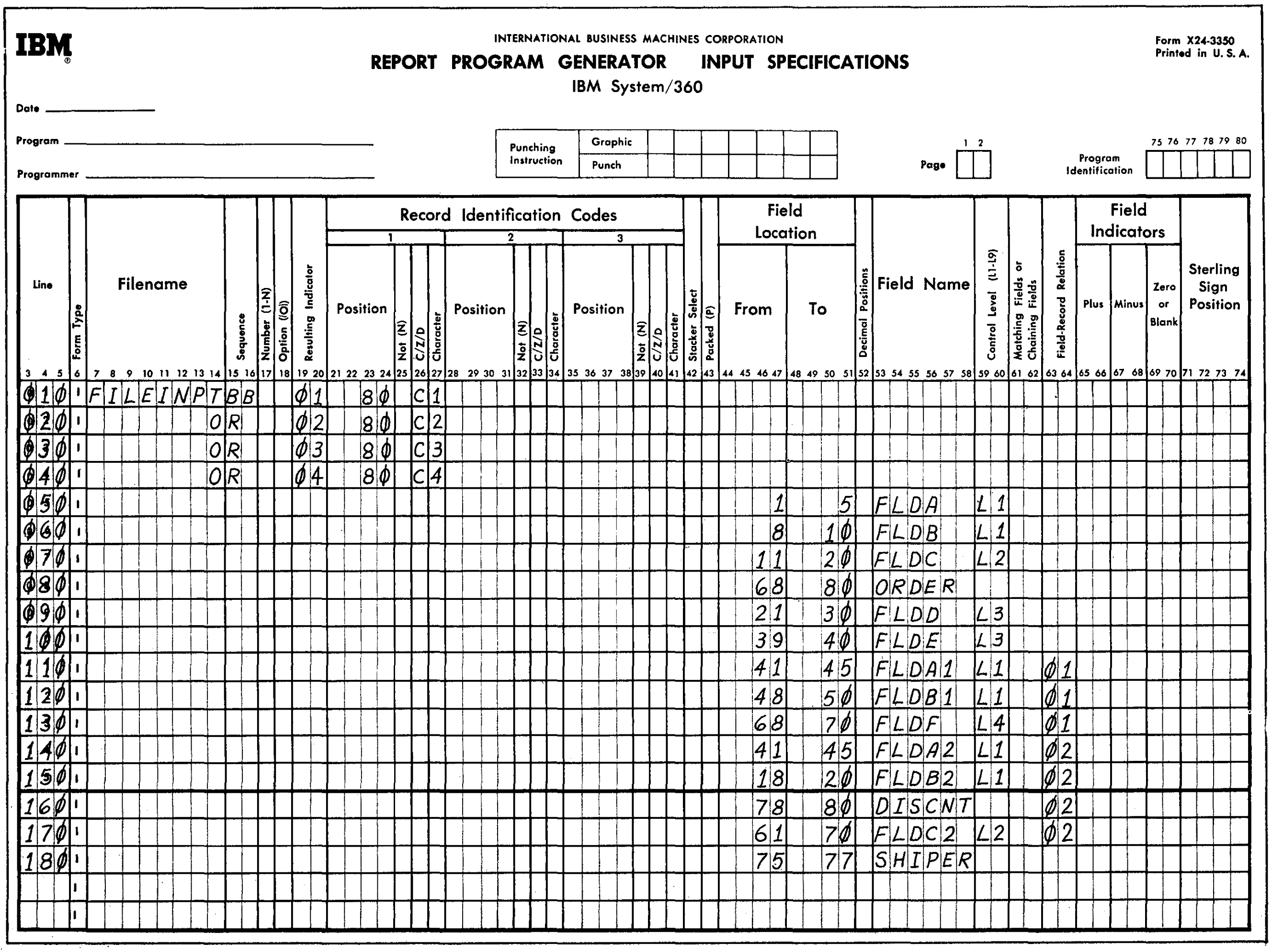
USING SPLIT CONTROL FIELDS WITH FIELD RECORD RELATION

Several rules must be observed when split control fields (columns 59-60) are specified in conjunction with Field-Record Relation (columns 63-64).

1. For each indicator, the sum of the length of the split fields must be equal to the length of the combined control fields as originally defined for a particular control level. In Figure 38, the sum of the lengths of FLDA and FLDB (L1 for blank indicator) must be equal to the sum of the lengths of FLDA1 and FLDB1 (L1 for indicator 01).

2. Within the specifications for a given indicator (in the Field-Record Relation columns), the control levels must be in ascending sequence. In Figure 38, the specifications for indicator 01 are entered in ascending sequence (L1, L1, and L4).

3. The split control levels specified with no entry in Field-Record Relation (blank resulting indicator) must be specified first. In Figure 38, the fields FLDA-FLDE (blank in columns 63-64) are entered before the remaining entries are specified.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date ________

Program ________

Programmer ________

Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

							Record Identification Codes														Field Location								Field Indicators			
							1				2				3																
Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Plus	Minus	Zero or Blank	Sterling Sign Position
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
ϕ1ϕ	I	FILEINPT	BB			ϕ1	8ϕ		C	1																					
ϕ2ϕ	I	O	R			ϕ2	8ϕ		C	2																					
ϕ3ϕ	I	O	R			ϕ3	8ϕ		C	3																					
ϕ4ϕ	I	O	R			ϕ4	8ϕ		C	4																					
ϕ5ϕ	I																				1	5		FLDA	L1						
ϕ6ϕ	I																				8	1ϕ		FLDB	L1						
ϕ7ϕ	I																				11	2ϕ		FLDC	L2						
ϕ8ϕ	I																				68	8ϕ		ORDER							
ϕ9ϕ	I																				21	3ϕ		FLDD	L3						
1ϕϕ	I																				39	4ϕ		FLDE	L3						
11ϕ	I																				41	45		FLDA1	L1		ϕ1				
12ϕ	I																				48	5ϕ		FLDB1	L1		ϕ1				
13ϕ	I																				68	7ϕ		FLDF	L4		ϕ1				
14ϕ	I																				41	45		FLDA2	L1		ϕ2				
15ϕ	I																				18	2ϕ		FLDB2	L1		ϕ2				
16ϕ	I																				78	8ϕ		DISCNT			ϕ2				
17ϕ	I																				61	7ϕ		FLDC2	L2		ϕ2				
18ϕ	I																				75	77		SHIPER							
	I																														
	I																														

Figure 38. Using Field-Record Relation

4. The split control fields associated with a specific indicator must be specified together. In Figure 38, the control levels associated with indicator 01 are specified together, and the control levels associated with indicator 02 are specified together.

5. Split control fields of the same level must not be separated. For example, it would be incorrect to place a data field between FLDA2 and FLDB2 (both L1). However, it is permissible to place a data field between different control levels. For example, the field ORDER is between FLDC (L2) and FLDD (L3).

6. If split control fields of one level in one type of record are specified with and without field-record relation indicators, only the fields conditioned by a field-record relation are valid whenever the appropriate indicator is on. The control fields without a field-record relation are disregarded.

7. Each split control field must be either alphabetic or numeric throughout.

MATCHING FIELDS OR CHAINING FIELDS (COLUMNS 61-62)

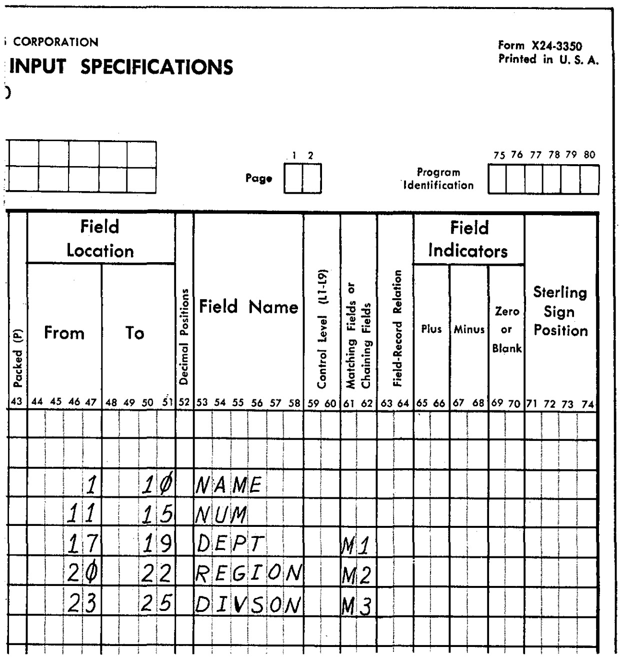
These columns are normally used if the input consists of more than one file. It enables the program to match or chain the records of one file with those of another file.

Matching Fields

If a field specified in Field Location also has an entry in Matching Fields, the field is placed into another main storage area known as a matching-field holding area.

Up to three matching fields (designated by M1-M3) are allowed. This entry may be used to match records in different sequential files. The second sample program at the back of this publication uses matching fields in a card input file and a tape input file to govern processing of records. A discussion of this use of matching fields may be found in Processing Multiple Input Files.

USING THE MATCHING FIELDS SPECIFICATION FOR SEQUENCE-CHECKING: If only one input file is specified, fields within the file may be sequence-checked by using the matching-fields specification. Up to three fields within the file may be checked. A chaining file may also be sequence-checked. If the chaining field is to be sequence-checked, it must be defined twice, using two different field names. In Figure 39, three fields within the input file are to be sequence-checked. The data from the three fields will be moved by the RPG program to the matching-field holding area as shown in Figure 40. When the second record has been read, it will be moved as shown. The compare operation is made on all three fields at the same time. M3 is placed in the highest-order position. M1 is placed in the low-order position.



i CORPORATION

INPUT SPECIFICATIONS

Form X24-3350
Printed in U. S. A.

Page 1 2

Program Identification 75 76 77 78 79 80

Packed (P)	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators Plus	Field Indicators Minus	Field Indicators Zero or Blank	Sterling Sign Position
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
	1	1Ø		NAME							
	11	15		NUM							
	17	19		DEPT		M1					
	2Ø	22		REGION		M2					
	23	25		DIVSON		M3					

Figure 39. Using Matching Fields in a Single Input File

Data Contained in First Record

DEPT	008
REGION	051
DIVSN	003

Data Contained in Second Record

DEPT	003
REGION	025
DIVSN	005

M3 DIVSN	M2 REGION	M1 DEPT	
003	051	008	Record 1

M3 DIVSN	M2 REGION	M1 DEPT	
005	025	003	Record 2

Figure 40. Comparing Matching Fields

In column 18 of the File Description Specification sheet the programmer must specify if the file is in ascending or descending sequence. In Figure 39 assume that the file has been specified in ascending sequence. The number 003051008 is lower than 005025003. Thus, the file is in ascending sequence.

EXIT TO A TRANSLATE SUBROUTINE

If the sequence of the matching fields is not the same as the collating sequence of the System/360, the RPG program can provide an automatic exit to an external user subroutine that translates the sequence of the matching fields to the collating sequence of the system.

An entry in the Processor Control Card is all that is required to cause the RPG program to branch to the subroutine. The automatic branch occurs after the input card is read in and before the RPG program checks the sequence of the matching field.

The subroutine to translate the matching fields must use the predefined label ALTSEQ. The register conventions for this subroutine are the same as those for the EXIT operation code. (See Exit to a Subroutine.) The address of the hold area containing the matching fields to be translated will be contained in Register 1. The subroutine must place the translated fields back into the matching-field holding area before it returns control to RPG.

The original data fields (to be used for controlling, printing, arithmetic calculations, etc.) are not translated by the subroutine.

Chaining Fields

The use of chaining files is explained in the section Chaining. Up to three chaining fields are permitted in a record. Chaining fields cannot be in packed decimal format and be specified as numeric. They must be specified as alphameric fields when packed. In these columns, enter the code that identifies the chaining field C1, C2, or C3.

FIELD-RECORD RELATION (COLUMNS 63-64)

This specification is used to specify records in an OR-relationship when the field records are not in the same location. Enter in columns 63-64 the appropriate resulting indicator which will be on when the field is used. An explanation of the use of this specification was contained in Records in an OR-relationship, and an example of this specification is in Figure 36.

Using Field-Record Relation with Chaining Files

These columns (63-64) can be used to condition chaining to a specific file. If the indicator entered in columns 63-64 is on when the chaining technique is used, the object program will then get the record from the designated file. When this column is blank, the chained record will be obtained whenever the specified record-type is present.

For example, if MR is entered in Field-Record Relation, the field will be used as a chaining field only if the matching record indicator is on.

If a field has an indicator (such as 01) specified in columns 65-70 which is used for conditional chaining, this field must precede the chaining field (which has the same indicator specified in Field-Record Relation).

FIELD INDICATORS (COLUMNS 65-70)

Entries in columns 65-70 are used to test the status of a field when it is read into the system. This entry is an indicator that will be turned on when the field specified on the line is plus, minus, or zero or blank.

Plus (Columns 65-66)

A plus condition occurs when the value of a numeric field is greater than 0.

Minus (Columns 67-68)

A minus condition occurs when the value of the field is less than 0.

Zero or Blank (Columns 69-70)

A zero condition occurs if a field contains all zeros or all blanks.

Types of Indicator Codes for Plus, Minus, and Zero or Blank

If the status of a field is tested so that calculation and output specifications can be modified, a two-digit field indicator is used in the appropriate columns (65-66, 67-68, or 69-70). These codes, from 01-99, can be defined one or more times on the Input Specification sheet. If the same indicator is defined on more than one specification, the last specification executed will determine the status of that indicator.

Using Figure 37, if the input-field BALNCE is a plus number, field-indicator 01 will be turned on. Indicator 01 will remain on until a new BALNCE is read in and tested. If FIELD2 is a minus number, field-indicator 02 will be turned on. If AMTDUE is zero or blank, field-indicator 03 will be turned on.

> Note: The indicator that is turned on remains on until the next time the field is tested by this particular specification.

Note that defining these indicators means that they have been entered in appropriate columns of the Input Specification sheet. Using these indicators means that they have been specified on the calculation or output sheets and that they are tested to determine their status. The field indicators will be discussed with the output and calculation sheets.

HALT INDICATORS

Nine additional indicator codes, known as halt indicators, can be used in the RPG program. These indicators, designated as H1 through H9, halt the processing of the object program when error conditions (as determined by the programmer) have been detected. These indicators can also be used in calculation and output specification sheets to control specifications and stop processing. For example: the status of a field can be tested, and depending upon the results of the tests, a halt indicator may be set on which will terminate the object program.

If one of these indicators has been turned on during the processing of a record, the object program is stopped at the completion of the processing of that record. However, processing will not be interrupted if a halt indicator that has been turned on is turned off before the program attempts to read the next input record.

STERLING (COLUMNS 71-74)

Enter in these columns the position in the record that contains the sign of the Sterling field. If the sign is in the normal position, enter an S in column 74. Leading zeros may be omitted. Leave these columns blank if the Sterling Specification is not used. Additional information on Sterling is in the section Sterling Routines for the Report Program Generator at the end of this publication.

SUMMARY OF INPUT SPECIFICATIONS

This concludes the description of the input specifications. Additional information on matching or chaining fields may be found in Processing Multiple Input Files. The input specifications listed are used with the calculation and output forms. The use and function of these specifications may become more apparent to the reader after the descriptions of the calculation and output specifications have been read:

Resulting Indicator
Field Name
Control Level
Matching Fields
Field Indicators.

This sheet is used to specify the operations to be performed by the object program upon the input data and upon data obtained as a result of other calculations. Two general rules govern the writing of calculation specifications:

1. Each operation is specified on one line of the sheet. Operations must be listed in the order in which they are to be performed in the object program.

2. Detail calculations must precede all total calculations. Calculations within these two groups must be specified in the order in which they are to be performed on the data.

The calculation sheet is divided into three categories as shown in Figure 41:

1. When calculations are to be performed. These entries (columns 7-17) determine when the calculations are to be performed and upon what conditions they are to be performed.

2. What kind of calculations are to be performed. These entries (columns 18-53) determine the kind of calculations to be performed, such as add, subtract, multiply, etc. These entries also supply information about the result of the calculation.

3. What tests are to be made on the results of the calculations. These entries (columns 54-59) test the results of the calculations to modify subsequent calculations or output specifications.

WHEN CALCULATIONS ARE TO BE PERFORMED (COLUMNS 7-17

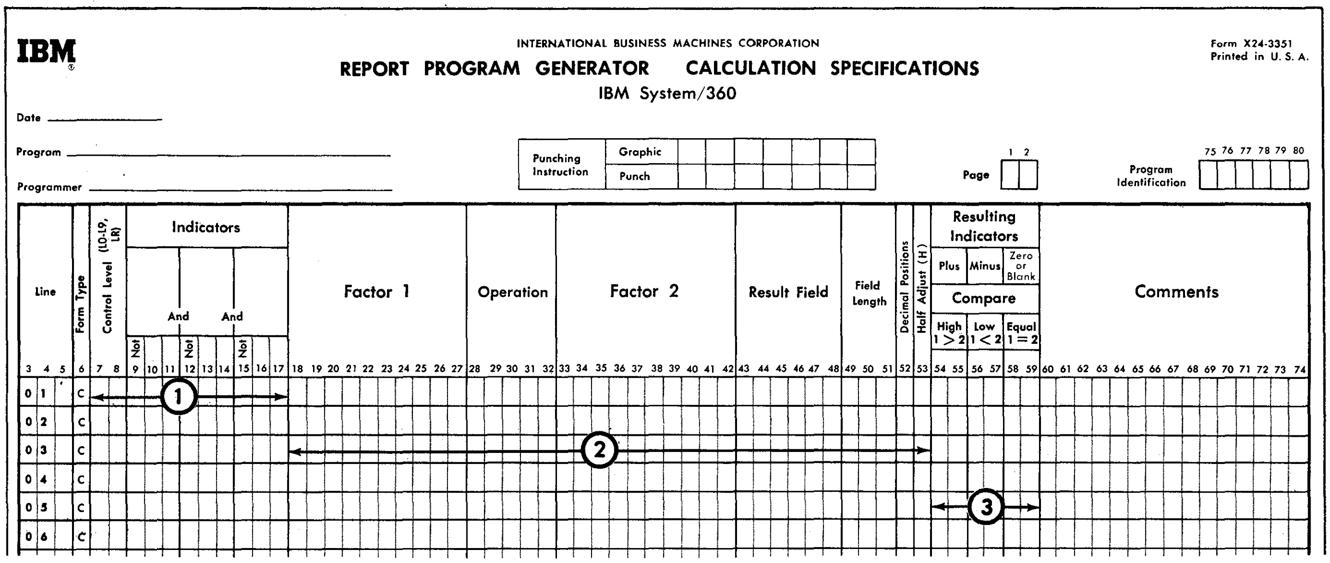
The two specifications in this category are control level and indicators.

CONTROL LEVEL (COLUMNS 7-8)

Calculations are performed either at detail time or total time. These columns (7-8) indicate the control level of calculations performed at total time. An entry (L1-L9, L0, LR) in these columns determines the control level of the calculation specified in columns 18-59.

A control break for a specific control level forces all lower control level indicators to be turned on. For example, an L3 break causes L3, L2, and L1 indicators to be turned on. A blank entry specifies that the calculation will be performed at detail time.

A test for a control change is made after each record is read into the system. Total calculations are performed after this test is made and before the record that caused the control change is processed.



IBM®

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction	Graphic
	Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Control Level (L0-L9, LR)	Indicators: Not	And: Not	And: Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Compare: High 1>2	Minus / Low 1<2	Zero or Blank / Equal 1=2	Comments
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
0 1	C															
0 2	C															
0 3	C															
0 4	C															
0 5	C															
0 6	C															

Figure 41. Calculation Specification Sheet

The entries for this specification are the control-level indicators L1-L9 and the indicators LR and L0. A control level indicator is turned on by a control break. It is on during total time, and it remains on for the following detail time, which includes both the calculating and the printing of the detail record. An entry of L1-L9 in these columns is obtained from the Control Level field on the Input Specification Sheet, columns 59-60.

LR (Last Record) Indicator

This indicator is turned on after the last input record has been read and calculated, and after the appropriate detail records are written. At this time, the control-level indicators L1 through L9 are also turned on.

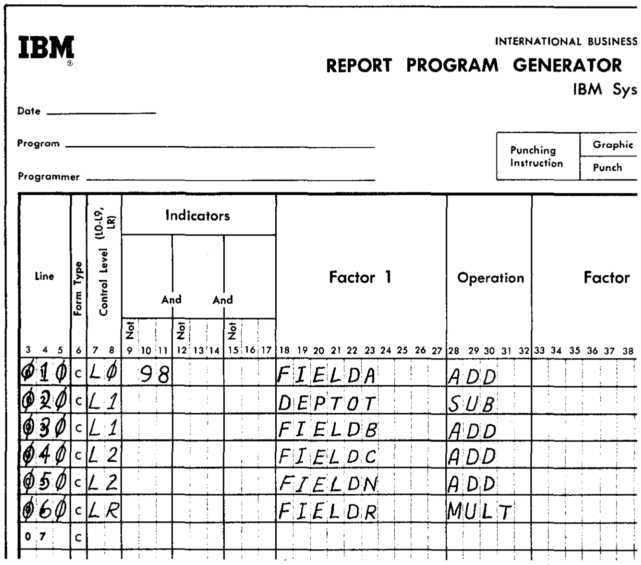
L0 (Level Zero) Indicator

This indicator is always on during the processing of the object program. By specifying L0, total calculations can be performed even though a control break has not occurred. This operation would be useful in an application when totals for one page of a printed report were to be accumulated when no control break had occurred.

Using Control Level Specifications

Figure 42 illustrates six control-break specifications. If an object program had only detail calculations, this specification would be left blank. The indicators that control detail calculations are specified in columns 9-17.

INDICATORS (COLUMNS 9-17)



IBM

INTERNATIONAL BUSINESS

REPORT PROGRAM GENERATOR

IBM Sys

Date

Program

Programmer

Punching Instruction	Graphic
	Punch

Line 3 4 5	Form Type 6	Control Level (L0-L9, LR) 7 8	Indicators Not 9	 10 11	And Not 12	 13 14	And Not 15	 16 17	Factor 1 18 19 20 21 22 23 24 25 26 27	Operation 28 29 30 31 32	Factor 33 34 35 36 37 38
ϕ1ϕ	C	Lϕ		98					FIELDA	ADD	
ϕ2ϕ	C	L1							DEPTOT	SUB	
ϕ3ϕ	C	L1							FIELDB	ADD	
ϕ4ϕ	C	L2							FIELDC	ADD	
ϕ5ϕ	C	L2							FIELDN	ADD	
ϕ6ϕ	C	LR							FIELDR	MULT	
0 7	C										

Figure 42. Using Control Level Specifications

The entries in columns 9-17 indicate the conditions that control the calculations specified in columns 18-59. From one to three indicator codes (considered to be in AND relationship) may be specified. The AND relationship means that if three indicator codes are specified, all three indicator conditions must be satisfied before the calculation can take place. Enter in columns 10-11, 13-14, and 16-17 the indicators that must be checked when the calculation is to be performed. If an indicator must be off, enter an N in either column 9, 12, or 15 (whichever is appropriate).

The specifications used in these columns can be arranged into the following categories:

1. If columns 9-17 and columns 7-8 are blank, the calculation specified on the line is performed each time a detail record is read.

2. A resulting-indicator code determines the particular record type on which the calculation is to be performed. This calculation will not be performed on any other record type.

3. A field-indicator code controls the calculation according to the status of an input field.

4. A resulting-indicator code controls the calculation by conditions that occurred on previous calculations. (This feature is shown on the Calculation Specification sheet, columns 54-59, but it has not been discussed at this time.)

5. A control-level indicator, L1-L9, used with a particular resulting indicator permits the calculation to be performed

at detail time, but it is performed only on the first record of the control level specified.

6. The matching-record indicator code, MR, means that the calculation is performed only if there is a matching record in a second input file.

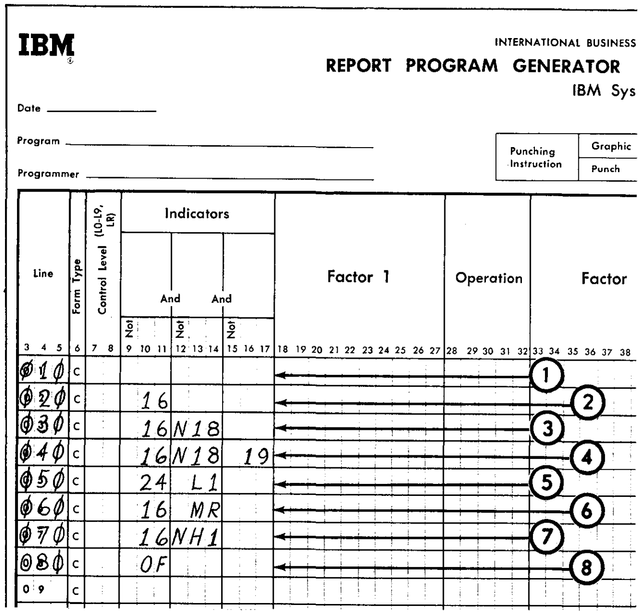
7. The halt indicators H1-H9 are normally used to terminate the program or to suppress a calculation when an error has been detected in the input data or on a previous calculation.

8. The overflow indicators permit the calculation to take place only if a page overflow has occurred.

9. All total calculations will be bypassed until the first control break has occurred.

In addition, this specification entry (columns 9-17) may contain a combination of some of the preceding categories. Also, a calculation may be controlled by the fact that an indicator must not be on.

Figure 43 shows entries that may be made in columns 9-17. The numbers to the right of the entries refer to the following list:



IBM

INTERNATIONAL BUSINESS
REPORT PROGRAM GENERATOR
IBM Sys

Date

Program

Programmer

Punching Instruction	Graphic
	Punch

Line	Form Type	Control Level (L0-L9, LR)	Indicators Not	And Not	And Not	Factor 1	Operation	Factor
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
Ø1Ø	C							①
Ø2Ø	C		16					②
Ø3Ø	C		16	N18				③
Ø4Ø	C		16	N18	19			④
Ø5Ø	C		24	L1				⑤
Ø6Ø	C		16	MR				⑥
Ø7Ø	C		16	NH1				⑦
Ø8Ø	C		OF					⑧
0 9	C							

Figure 43. Using Indicators

1. The first example is a blank entry. This means that the calculation is performed on every detail record.

2. In this example, indicator 16 could be a resulting indicator for a specific record-type.

3. In this example, indicator 16 could be a resulting indicator, and indicator 18 could be a field indicator, which is specified on the Input Specification sheet. If indicator 18 is used to test an input field for blanks, this entry means, in effect, that the calculation would be performed if record-type 16 is present and the contents of the field represented by indicator 18 is not blank.

4. This example is similar to the previous example. However, indicator 19 could be a resulting indicator, turned on by the previous calculation. The calculation specified on this line in columns 18-59 would not be performed unless indicator 19 was also on.

5. This entry means that the calculation is performed at detail time on control-level 1, and only if indicator 24 is on.

6. This entry means that the calculation is performed only if indicator 16 is on and there is a matching record condition. For example, when fields from detail records are multiplied by a factor contained in a master record, the program must have a way of insuring that the detail record has been matched with the appropriate master record.

7. The entry of H1 prevents the object program from performing the calculation if an error condition has occurred. Note that when an error occurs, the job is not terminated until after all processing for the record has been completed. This facility is provided so that the programmer can prevent the calculation of erroneous data.

8. This entry means that the calculation is performed on the detail cycle following an overflow condition.

KINDS OF CALCULATIONS TO BE PERFORMED

This section describes the kind of calculations to be performed. These specifications answer the following three questions:

1. What fields are to be used?

2. What is the operation (add, subtract, multiply, etc.)?

3. What is done with the result?

FACTOR 1 (COLUMNS 18-27)

This specification can be a field name or a literal. If it is a field name it must have been defined on the Input Specification sheet, or it must have been defined in the result field of another calculation. The field name must be left-justified, and the first character must be either alphabetic or @, #, or $.

Literals

A literal is the actual data to be operated on, rather than a name representing the location of data. Literals may be numeric or alphabetic. Numeric literals may be up to ten characters long. Alphameric literals may be up to eight characters long.

NUMERIC LITERALS: A numeric literal must consist of any combination of the digits 0-9. A numeric literal can also have specified a leading plus or minus sign and/or one decimal point.

Rules for Forming Numeric Literals:

1. Blanks must not appear within a numeric literal.

2. The sign, if present, must be the leftmost character. If a literal is unsigned, it is treated as a positive literal.

3. The decimal point can appear anywhere in the literal.

4. Numeric literals must not be enclosed in quote symbols.

ALPHAMERIC LITERALS: Any set of consecutive characters enclosed in a set of single-quote symbols is treated as an alphameric literal. Alphameric literals may be used for compare operations, but they may not be used in computations.

Rules for Forming Alphameric Literals:

1. Any character may be used in an alphameric literal. Blanks are treated as valid characters in the body of the literal.

2. Alphameric literals must be enclosed in a set of single-quote symbols.

3. A single quote symbol may be contained within a literal by entering an additional single quote symbol within the literal. For example, the literal o'clock would be coded as 'o''clock'.

Figure 44 illustrates entries for Factor 1.

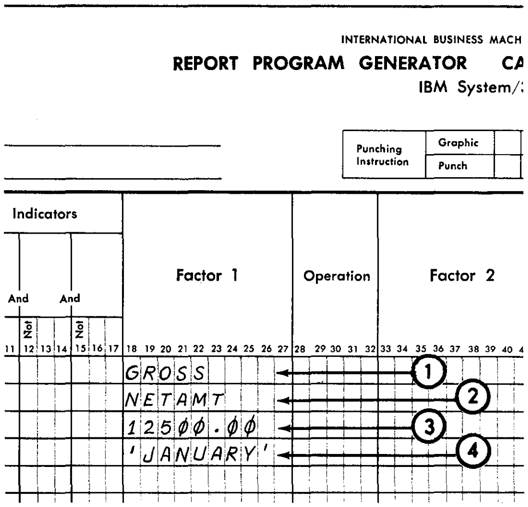


Figure 44. Factor 1

1. GROSS could be a field name specified on the input sheet.

2. NETAMT could have been specified as the result field of the previous calculation.

3. This numeric literal could be used in the program to determine if specific fields in the input records were higher or lower than this number. The decimal position must be indicated (if the number is not a whole number) but commas must not be used.

4. Alphameric literals like the one in this example can be used to compare against a data field in the input records to perform certain types of calculations, only upon records representing the month of January.

The discussion of Factor 1 also applies to Factor 2.

SUMMARY OF FACTOR 1 AND FACTOR 2

1. Enter either the name or the literal that is Factor 1 or Factor 2 in columns 18-27, or 33-42.

2. If Factor 1 or Factor 2 contains the name of a field, the field must be defined in either:

   a. Columns 53-58 (Field Name) of the Input Specification sheet.

b. Columns 43-48 (Result Field) of the Calculation Specification sheet.

3. A name cannot exceed six characters. Special characters and blanks must not be used.

4. A literal cannot exceed ten characters.

5. Entries in Factor 1 or Factor 2 must be left-justified.

OPERATION (COLUMNS 28-32)

Entries in these columns specify the operation to be performed using Factor 1, Factor 2, and the Result Field. Each operation is specified by placing the operation code in the operation field (Columns 28-32).

Arithmetic Operations	Code
Add	ADD
Zero and Add	Z-ADD
Subtract	SUB
Zero and Subtract	Z-SUB
Multiply	MULT
Divide	DIV
Move Remainder	MVR

Move Operations	Code
Move	MOVE
Move-Leftmost-Data-Characters	MOVEL
Move Low-to-High Zone	MLHZO
Move High-to-Low Zone	MHLZO
Move High-to-High Zone	MHHZO
Move Low-to-Low Zone	MLLZO

Testing or Compare Operations	Code
Compare	COMP
Test Zone	TESTZ

Branching and Exit Operations	Code
Branching (or Go To)	GOTO
Providing a Label for GOTO ..	TAG
Exit to a Subroutine	EXIT
RPG Label	RLABL
User Label	ULABL

Turning Indicators ON or OFF	Code
Set Indicators On	SETON
Set Indicators Off	SETOF

Table Operations	Code
Table Lookup	LOKUP

Conversion Routine Operations	Code
External Conversion Routine .	EXTCV
Record Key	KEYCV

ARITHMETIC OPERATIONS

The fields or literals involved in this operation may contain numeric characters only. All arithmetic operations are performed with automatic decimal alignment. No arithmetic overflow will be sensed by RPG. The length of a field involved in arithmetic operations can be up to 15 digits (this includes decimal alignment when necessary). The resulting field length after decimal alignment must not be greater than 15 digits.

ADD (ADD)

This operation causes the contents of the field or the literal in Factor 2 to be added, algebraically, to the contents of the field or literal in Factor 1. The result of the operation is placed in the result field specified in columns 43-48.

ZERO AND ADD (Z-ADD)

This operation causes the result field to be set to zeros and then causes the data contained in the numeric literal or the field in Factor 2 to be placed in the result field. Factor 1 is not used in this operation.

SUBTRACT (SUB)

This operation causes the contents of the field or literal in Factor 2 to be subtracted, algebraically, from the contents of the field or literal in Factor 1. The result of this operation is placed in the result field specified.

ZERO AND SUBTRACT (Z-SUB)

This operation causes the negative of the number contained in the literal or the field in Factor 2 to be placed in the result field specified. This operation is performed after the result field has been set to zeros. Factor 1 is not used in this operation.

MULTIPLY (MULT)

This operation causes the contents of the field or literal in Factor 1 to be multiplied algebraically by the contents of the field or the literal in Factor 2. The result of this operation is placed in the result field specified.

DIVIDE (DIV)

This operation causes the contents of the field or literal in Factor 1 to be divided by the contents of the field or literal in Factor 2. The result of this operation (quotient) is placed in the specified result field. The contents of the field or the literal in Factor 2 cannot be zero. If Factor 2 is zero, a program check and an abnormal end-of-job will result. Any remainder that results from this operation is lost unless the move-remainder operation is specified as the next operation.

> Note: If a move remainder operation follows a divide operation, the result in the divide operation cannot be half-adjusted.

In this operation, RPG performs the adjustment of fields (divisor and dividend) when necessary. The divisor and dividend are adjusted by adding zeros to the right of the units position of that field. The following formula determines whether or not field adjustment is necessary:

A = (Number decimal positions of Result Field) + (Number decimal positions of Factor 2) - (Number decimal positions of Factor 1)

If A=O, no adjustment takes place.

If A>O, the dividend will be adjusted.

If A<O, the divisor will be adjusted.

The amount of adjustment (the number of zeros added to the right of the units position) is determined by the absolute value of A in the preceding formula.

MOVE REMAINDER (MVR)

This operation is used to move the remainder from a divide operation to a separate field that has been set to zero by the RPG processor. If MVR is used, it must follow the divide operation. Figure 45 shows an example of the MVR operation. The remainder is placed in a field named STORE. The field that contains the remainder must be specified in the result field.

The value of the remainder can be determined by the following formula:

R = Dividend - (Divisor) (Quotient)

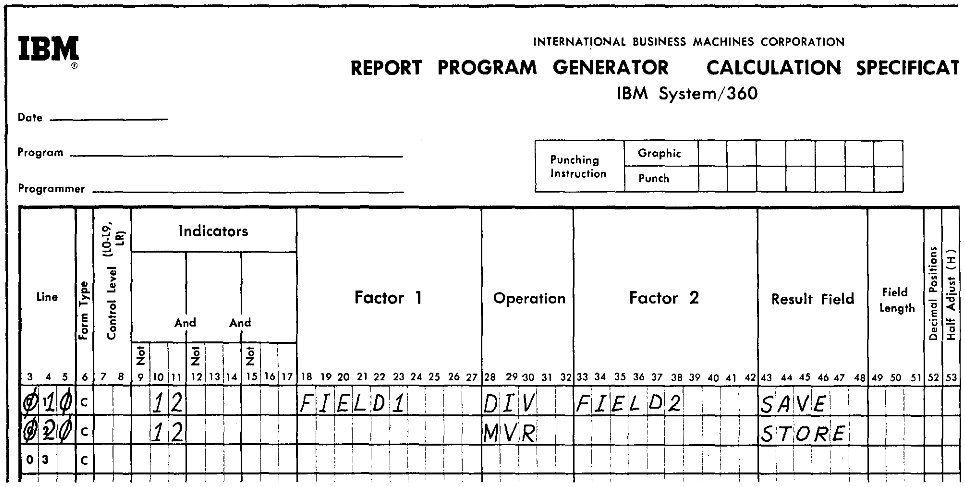
The decimal positions of the remainder following the DIV operation are equal to the dividend decimal length (adjusted).

> Note: A move remainder operation can specify resulting indicators.

MOVE OPERATIONS

For the MOVE and MOVEL operations, unpacked numeric fields may be changed to alphameric fields, and packed alphameric fields may be changed to numeric fields. To change a numeric field to an alphameric field, Factor 2 is numeric, and the result field is specified as alphameric. To change an alphameric field to a numeric field, Factor 2 is alphameric, and the result field is specified as numeric. No decimal alignment is performed when a move operation is used. When moving an alphameric field to a numeric field, a validity check is not performed on the sign of the result field. The result field will contain the standard plus or minus sign.

> Note: A move operation cannot specify resulting indicators.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICAT

IBM System/360

Date

Program

Programmer

Punching Instruction	Graphic							
	Punch							

Line	Form Type	Control Level (L0-L9, LR)	Indicators Not		And Not		And Not		Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)
Ø1Ø	C			12					FIELD1	DIV	FIELD2	SAVE			
Ø2Ø	C			12						MVR		STORE			
03	C														

Figure 45. Using the MVR Operation Code

MOVE (MOVE)

This operation code causes data characters (starting at the rightmost position) to be moved from the field or literal contained in Factor 2 to the rightmost positions of the result field.

If Factor 2 is longer than the result field, the excess leftmost positions of Factor 2 are not moved.

If the result field is longer than the field specified by Factor 2, the positions to the left of the data that is moved remain undisturbed.

Factor 1 is not used in this operation.

MOVE LEFT (MOVEL)

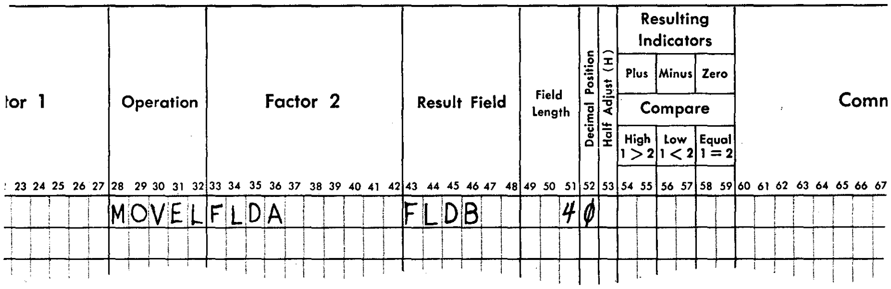
This operation code causes (starting at the leftmost position) data characters to be moved from the field or literal contained in Factor 2 to the leftmost positions of the result field.

If Factor 2 is longer than the result field, the excess rightmost positions of Factor 2 are not moved.

If the result field is longer than the field specified by Factor 2, the positions to the right of the data that is moved remain undisturbed. When moving data to a numeric field, the sign of the result field is retained, except when Factor 2 is as long or longer than the result field. In this case, the sign of Factor 2 is assumed.

Factor 1 is not referenced in this operation.

Figure 46 shows a use of this operation code. FLDA contains a five-position number (10000). FLDB is a four-position field. The result of the MOVEL is 1000 in FLDB. In this example, the MOVEL operation code performs the same function as dividing FLDA by 10.



tor 1	Operation	Factor 2	Result Field	Field Length	Decimal Position	Half Adjust (H)	Resulting Indicators Plus / Compare High 1 > 2	Minus / Low 1 < 2	Zero / Equal 1 = 2	Comn
	MOVEL	FLDA	FLDB	4	Ø					

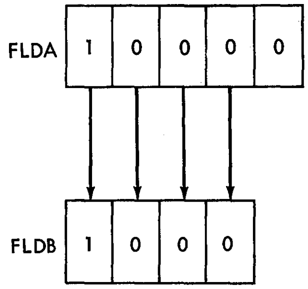


Figure 46. Using the MOVEL Operation Code

MOVE HIGH-TO-LOW ZONE (MHLZO)

This operation moves the zone at the leftmost position of Factor 2 to the rightmost position of the result field. Factor 2 can only be alphameric. If the zone to be moved is located over numeric data, this operation can still be performed; however, the numeric field must have been specified as an alphameric field on the Input Specification sheet. The result field can be numeric or alphameric. A result field specified as numeric always contains the standard plus or minus zone after this operation. Figure 47 illustrates this operation.

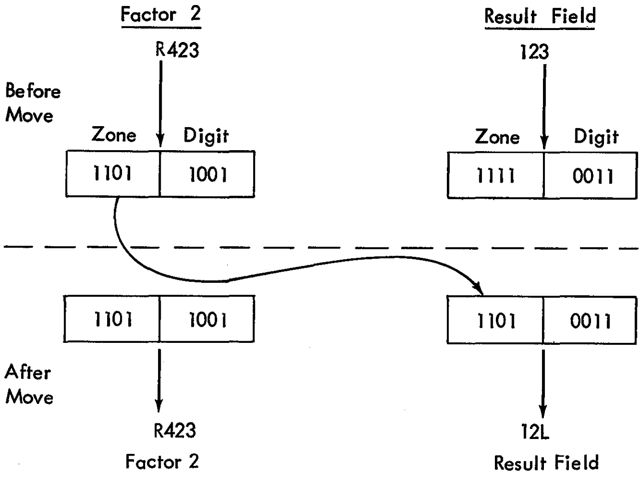


Figure 47. Move High-to-Low Zone Operation

MOVE LOW-TO-HIGH ZONE (MLHZO)

This operation moves the zone at the rightmost position of Factor 2 to the leftmost position of the result field. Factor 2 can be numeric or alphameric, but the result field must be alphameric.

MOVE HIGH-TO-HIGH ZONE (MHHZO)

This operation moves the zone at the leftmost position of Factor 2 to the leftmost position of the result field. Factor 2 and the result field must be alphameric.

MOVE LOW-TO-LOW ZONE (MLLZO)

This operation moves the zone at the rightmost position of Factor 2 to the rightmost position of the result field. Factor 2 and the result field are alphameric or numeric. A result field specified as numeric always contains the standard plus or minus zone after this operation.

Figure 48 illustrates the movement of zones for all of the preceding four operation codes.

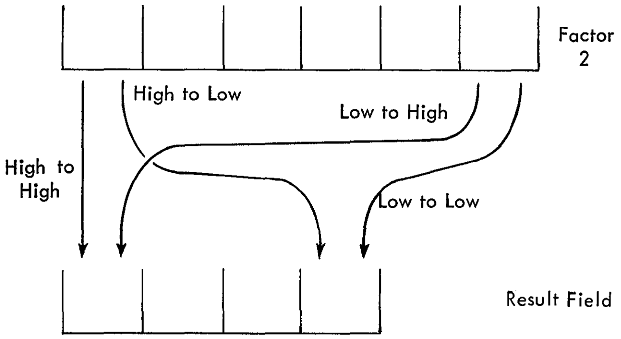


Figure 48. Move Zone Operation

TESTING OR COMPARE OPERATIONS

COMPARE (COMP)

This operation causes the contents of the field or the literal in Factor 1 to be compared against the contents of the field or literal in Factor 2. The outcome of this operation can be used to turn on an indicator that has been specified in columns 54-59 (Result Indicators High, Low, or Equal). These indicators are turned on as follows:

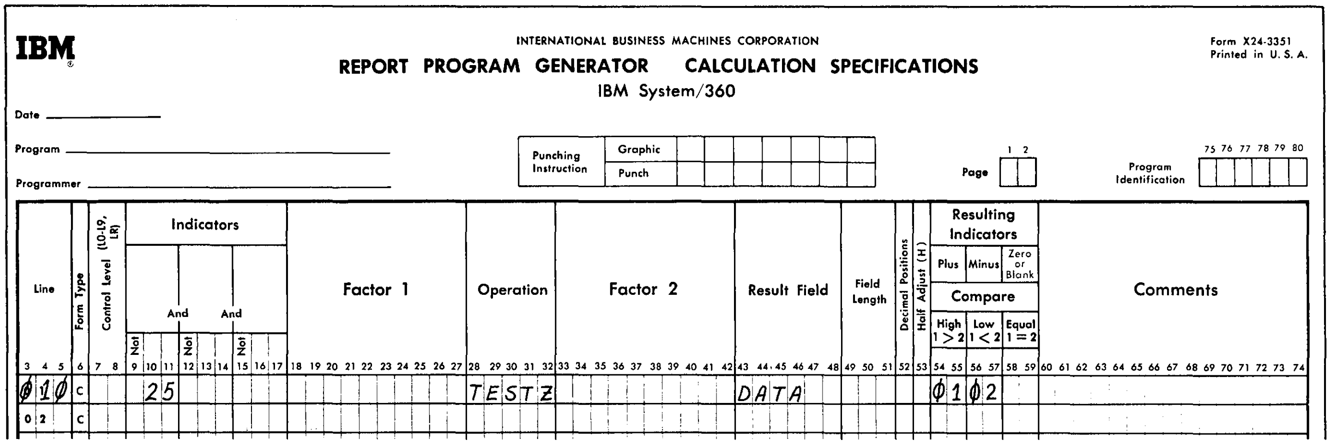
High — Factor 1 is greater than Factor 2.

Low — Factor 1 is less than Factor 2.

Equal — Factor 1 is equal to Factor 2.

This operation is used to make comparisons to alter or modify subsequent calculations. No result field is specified.

- The Factor-1 and Factor-2 fields are aligned according to whether they are numeric or alphameric. If numeric fields are compared, fields of unequal length are aligned to the implied decimal point.
- Missing digits in numeric fields are assumed to be zeros.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Control Level (L0-L9, LR)	Indicators	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Plus (High 1>2)	Minus (Low 1<2)	Zero or Blank (Equal 1=2)	Comments
Ø1Ø	C		25		TESTZ		DATA				Ø1	Ø2		
02	C													

Figure 49. Using TEST ZONE

- If alphameric fields are compared, fields of unequal length are aligned to their leftmost characters.

- The alphameric compare operation is based upon the internal collating sequence of the system.

- For numeric fields, the maximum field length is 15 digits.

- For alphameric fields of equal length, the maximum field length is 256 characters. For alphameric fields of unequal length, the maximum field length is 40 characters.

TEST ZONE (TESTZ)

This operation is used to test the zone of the leftmost position of the alphameric field that is entered in the result field. (The result field must be alphameric.) If the result of the test is a 12-zone (plus), the indicator specified in columns 54-55 will be turned on. If the result of the test is an 11-zone (minus), the indicator specified in columns 56-57 is turned on. Any other zone turns on the indicator specified in columns 58-59.

Figure 49 shows an example of this operation. When indicator 25 is on, the field DATA is tested. If the leftmost position has a 12-zone, indicator 01 is turned on. If the leftmost position has an 11-zone, indicator 02 is turned on.

If an ampersand is tested, it is considered as a 12-zone. A minus is considered an 11-zone.

BRANCHING AND EXIT OPERATIONS

This operation code enables the programmer to transfer control from the RPG program to a user subroutine. Factor 2 contains the name of the subroutine. See EXIT to a User's Routine for a complete discussion of this operation.

RPG LABEL (RLABL)

This operation provides the facility for a subroutine, external to the RPG program, to reference a field in the RPG program. The name of the field to be referenced is entered in the result field.

> Note: When Table Name is specified as an RLABL, the address supplied is that of the leftmost position of the table holding area.

USER'S LABEL (ULABL)

This operation enables the RPG program to reference a field contained in a user subroutine. The name of the field to be referenced is entered in the result field. The field length must be defined.

BRANCHING OR GO TO (GOTO)

The operation code GOTO enables branching to occur in the object program. This means leaving one point in the program to begin operations at some other location in the program. The location of the other routine is identified by a name. This name is entered in Factor 2 and the code GOTO is entered in Operation (columns 28-31). For example, a routine (a group of specifications) to calculate the employee contribution to the Federal Insurance Contribution Act might be labeled FICA. Branching to this routine would require that Factor 2 of the GOTO operation contains the name FICA and that the first operation in the routine be a TAG operation; that is, an operation that defines the name of the routine (refer to description of next operation code).

Branching can be performed within detail calculations or within total calculations, but not between detail and total calculations. Branching can be forward (skipping over specifications) or backward (going back to specifications previously skipped or processed). (For additional information, see Using the Calculation Specification sheet.)

PROVIDING A LABEL FOR GOTO (TAG)

The operation TAG (columns 28-30) provides a name to which the program can branch. Enter this name in Factor 1 and the code TAG in operation (columns 28-32). The name will be used as Factor 2 of the operation code GOTO. If the specifications in the TAG routine occur at total time, then the TAG operation must have L0 specified in Control Level. If they occur at detail time, a control specification is not required.

TURNING INDICATORS ON OR OFF

SET INDICATORS ON (SETON)

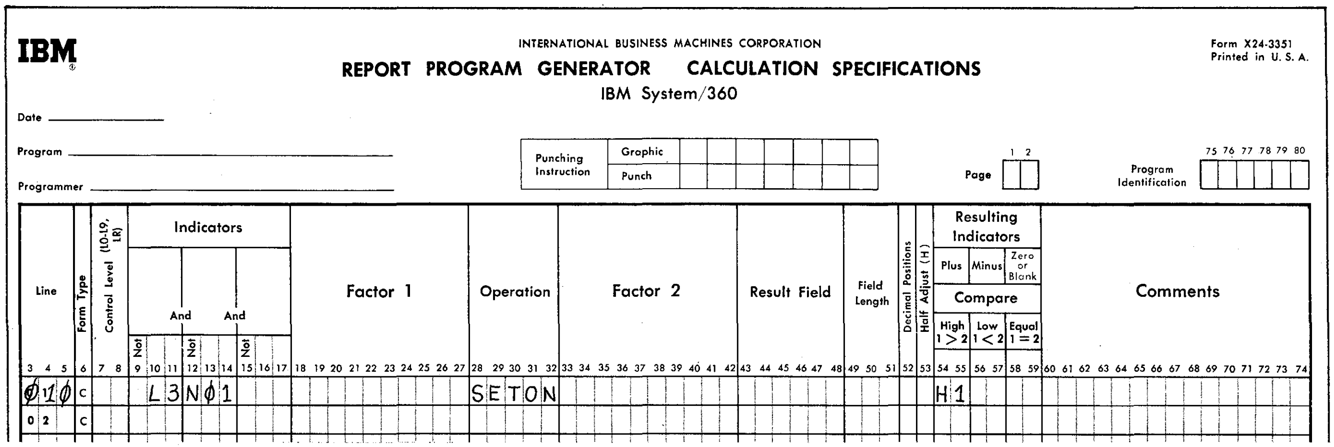
This operation code causes the indicators specified in columns 54-55, 56-57, or 58-59 to be turned on.

> Note: The column headings of plus, minus, or zero and high, low, or equal have no meaning during this operation, and should be ignored. Columns 54 through 59 are used for this operation code merely to record three sets of indicator codes.

Specify the first indicator in columns 54-55, the second indicator in columns 56-57, and the third indicator in columns 58-59. One use of this specification is to turn on a halt indicator when input records are out of sequence. Any RPG indicator may be turned on. Figure 50 shows an example of this facility. The 01 is an indicator that is set on for the first record of a sequence. The L3 is a control level that occurs with the first record of the sequence. If such a situation (L3 and 01) does not occur, the halt indicator H1 is turned on.

SET INDICATORS OFF (SETOF)

This operation code causes the indicators specified in columns 54-55, 56-57, or 58-59 to be turned off. Any RPG indicators may be turned off. When an L3 break occurs, L1-L3 are turned on. If the user did not want L1 on, he could have turned it off by using the SETOF operation code.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR) (7-8)	Indicators: Not (9)	(10-11)	And: Not (12)	(13-14)	And: Not (15)	(16-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators: Plus / Compare: High 1>2 (54-55)	Minus / Low 1<2 (56-57)	Zero or Blank / Equal 1=2 (58-59)	Comments (60-74)
Ø1Ø	C			L3	N	Ø1				SETON						H1			
02	C																		

Figure 50. Using SETON for a Record Out of Sequence

TABLE OPERATIONS

TABLE LOOKUP (LOKUP)

This operation code causes the field name contained in Factor 1 to be used as the search argument in a table lookup operation. Factor 2 contains the name of the table to be searched, and the result field contains the table from which the associated function will be obtained. Decimal alignment is not performed for this operation. (The result field may be left blank if the user wants to determine if an argument is present in the table, but does not require the corresponding function.)

After the lookup operation is completed, the function that is retrieved is placed in a special holding area within the function table. The name of this area is the same as the name of the function table. The retrieved data remains in this area until the next table lookup operation.

To utilize the function in another operation (for example in an arithmetic operation), the name of the function table is specified in Factor 1 or Factor 2. In this case, the function table name in Factor 1 or Factor 2 refers to the special holding area in the function table.

To update the function table (for example, a move operation to replace the old function with the new updated function), the name of the function table is specified in the Result Field. The new function is then placed in its proper place in the function table.and in the special holding area. In this case, the function table name in the result field refers to the location of the function within the function table and within the special holding area.

After each table lookup operation, the retrieved function should be used (or moved from the special holding area to another location) before the next table lookup operation is performed. Each subsequent lookup operation overlays the function obtained from the previous lookup operation.

See Using Tables in the Object Program for additional information and examples of table lookup operations.

CONVERSION ROUTINE OPERATIONS

EXTERNAL CONVERSION ROUTINES (EXTCV)

This operation is used to indicate the point in the RPG program where the random address conversion routine is to be performed. Factor 1 contains the internal RPG label as specified in columns 27-32 of the File Extension Specification sheet. Factor 2 must contain the label of the external address conversion routine. The result field contains the name of the field that will contain the track address. The track-address length is always eight characters.

> Note: All EXTCV operations must be specified as the initial entry(s) on the Calculation Specification sheet. Up to 4 of these operations are permitted.

RECORD KEY (KEYCV)

This operation code establishes the name of the field that is to contain the key of the disk record. (It is used only when records are retrieved using record key data.) The code KEYCV is placed in Operation (columns 28-32) and the name of the key field is placed in Result Field (columns 43-48). The field length and decimal positions must be specified.

If the records are retrieved by key field, this operation must follow its EXTCV operation. The name of the field that will contain the record key cannot be declared in the external conversion routine.

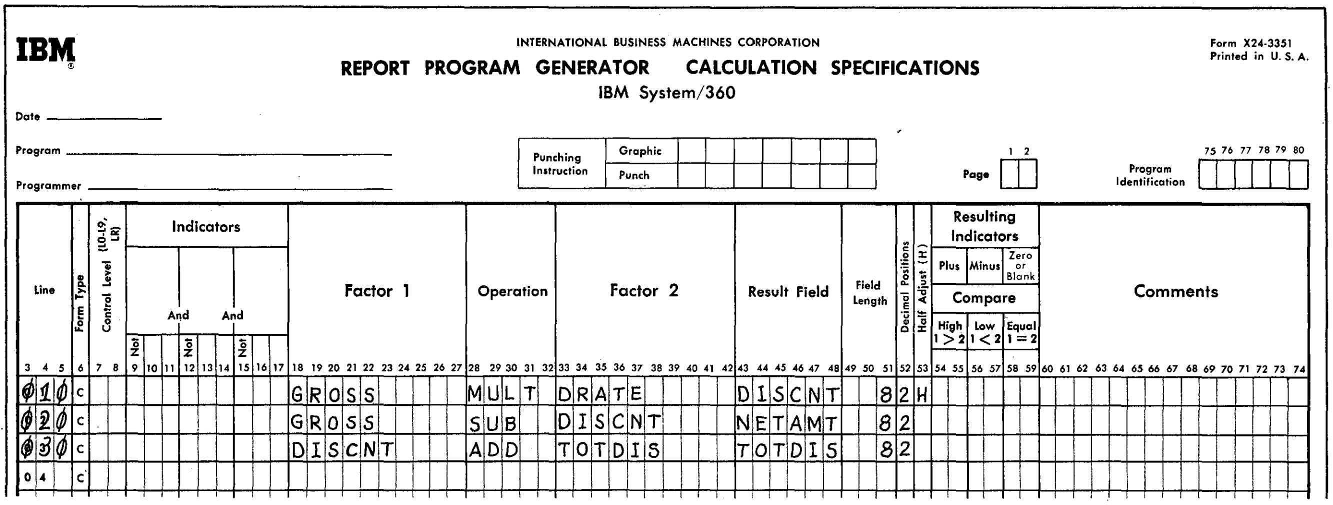
FACTOR 2 (COLUMNS 33-42)

The discussion of Factor 1 also applies to Factor 2.

RESULT FIELD (COLUMNS 43-48)

The name of the result field, which is entered in these columns, sets up a location in storage to contain the result of a calculation. The name of the result field must consist of alphameric characters, or @, #, or $, and must be left-justified.

The same name can be used several times in different calculations if the length of the field and the number of decimal positions are the same for all calculations.



IBM®

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date ______

Program ______

Programmer ______

Punching Instruction	Graphic							
	Punch							

Page (1 2) □□ Program Identification (75 76 77 78 79 80) □□□□□□

Line (3 4 5)	Form Type (6)	Control Level (L0-L9, LR) (7 8)	Indicators Not (9)	(10 11)	And Not (12)	(13 14)	And Not (15)	(16 17)	Factor 1 (18–27)	Operation (28–32)	Factor 2 (33–42)	Result Field (43–48)	Field Length (49 50 51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators Plus / Compare High 1 > 2 (54 55)	Minus / Low 1 < 2 (56 57)	Zero or Blank / Equal 1 = 2 (58 59)	Comments (60–74)
Ø1Ø	C								GROSS	MULT	DRATE	DISCNT	8	2	H				
Ø2Ø	C								GROSS	SUB	DISCNT	NETAMT	8	2					
Ø3Ø	C								DISCNT	ADD	TOTDIS	TOTDIS	8	2					
04	C																		

Figure 51. Result Field Entry

Figure 51 shows the result-field entry. On the first line GROSS is multiplied by DRATE, and the result field is DISCNT. DISCNT is then used as Factor 2 on the next line to calculate the net amount. The same result field is then used as Factor 1 on the next line to calculate total discount.

FIELD LENGTH (COLUMNS 49-51)

This entry specifies the length of the result field. The entry must specify the number of positions to be reserved for the result field. In Figure 51, DISCNT is eight positions long. The unpacked length must be specified. The maximum numeric field length is 15 digits, and the maximum alphameric field length is 256 characters.

If the same field name is used for more than one calculation and the field length and the number of decimal positions are the same, the field-length specification need not be specified more than once.

Note: If Result Field contains a table name, these columns must be left blank.

DECIMAL POSITION (COLUMN 52)

An entry in this column indicates the number of positions to the right of the decimal point in the result field. An entry must be made in this column for all arithmetic operations if the field has not been defined previously. If the result field does not have any decimal positions, the entry must be a 0. A maximum of nine decimal positions can be specified. In Figure 51 each result field has two decimal positions.

If the result field is alphameric, this column must be left blank.

HALF-ADJUST (COLUMN 53)

This specification is used to half-adjust, or round, the result field. Enter an H in this column if the data in the result field is to be half-adjusted.

Half-adjusting is accomplished in the object program by adding a 5 to the right of the last position retained in the result field.

In Figure 51, DISCNT is half-adjusted.

This completes the description of the specifications required for deᴗermining the kind of calculations to be performed.

TESTING THE RESULTS OF A CALCULATION

The last category of the Calculation Specification sheet is used to test the results of the calculation.

RESULTING INDICATORS (COLUMNS 54-59)

An entry in columns 54-59 may be used to test the value of a result field after the completion of an operation. As a result of the test, an indicator may be turned on that can be used to control subsequent operations or to control output operations. A result-field indicator must be specified when using a compare or a table-lookup operation.If the same indicator is defined on more than one specification, the last specification executed will determine the status of that indicator.

The entries in columns 54-59 are used in five ways:

1. To determine whether the result of an arithmetic operation is plus, minus, or zero.
2. To test the result of a compare operation.
3. To test the result of a table-lookup operation.
4. With the test-zone operation
5. With the set-indicators-on and set-indicators-off operations.

Result Field Indicator (Plus or High) Columns 54-55

Enter in these columns a two-digit indicator that will be set on when the result field, in an arithmetic operation, is a plus number.

If a compare operation was performed, the indicator is set on when Factor 1 is higher than Factor 2.

If a table-lookup operation was performed, enter the indicator that will be set on when the table argument next higher than the search argument is found.

Result Field Indicator (Minus or Low) Columns 56-57

Enter in these columns a two-digit indicator that will be set on when the result field, in an arithmetic operation, is a minus number.

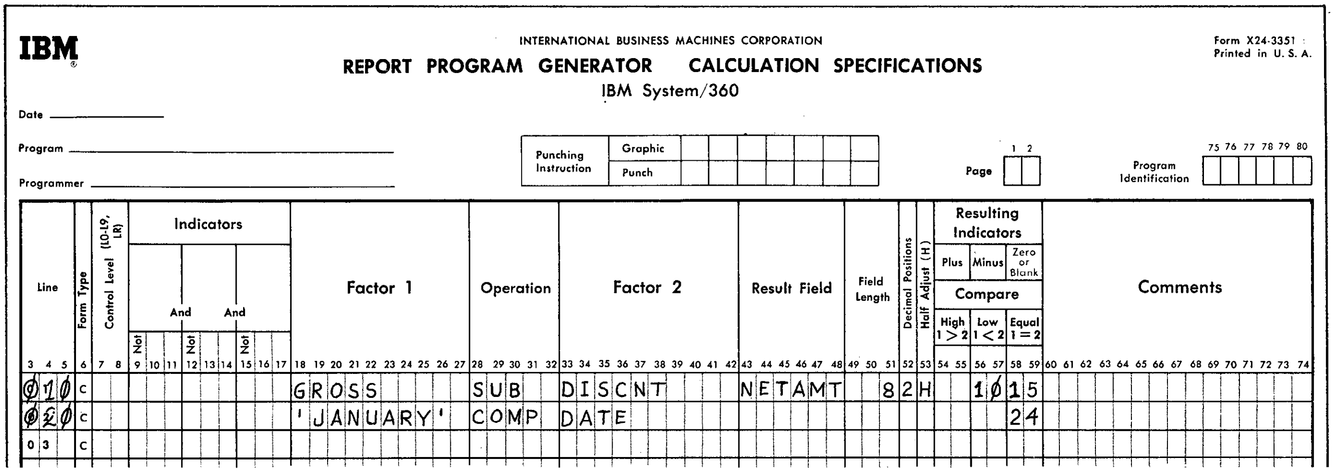
If a compare operation was performed, enter a two-digit indicator that will be set on when Factor 1 is lower than Factor 2.

If a table-lookup operation was performed, enter a two-digit indicator that will be set on when the table argument next lower than the search argument is found.

In Figure 52, DISCNT is subtracted from GROSS and the result is stored in NETAMT. If the answer is a minus number, indicator 10 is set on. If the answer is zero, indicator 15 is set on.

Result Field Indicator (Zero or Equal) Columns 58-59

Enter in these columns a two-digit indicator that will be set on when the result field, in an arithmetic operation, is zero.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction | Graphic | Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Control Level (L0-L9, LR)	Indicators Not	And Not	And Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators Plus / Compare High 1 > 2	Minus / Low 1 < 2	Zero or Blank / Equal 1 = 2	Comments
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
ø1ø	C					GROSS	SUB	DISCNT	NETAMT	8	2	H		1ø	15	
ø2ø	C					'JANUARY'	COMP	DATE							24	
0 3	C															

Figure 52. Result Field Indicators

If a compare operation was performed, enter a two-digit indicator that will be set on when Factor 1 is equal to Factor 2.

If a table-lookup operation was performed, enter a two-digit indicator that will be set on when the search argument is equal to an argument found in the table.

In Figu:e 52, the literal JANUARY is compared against the contents of DATE. If the result is equal, indicator 24 is turned on.

COMMENTS (COLUMNS 60-74)

These columns may be used for comments.

USING THE CALCULATION SPECIFICATION SHEET

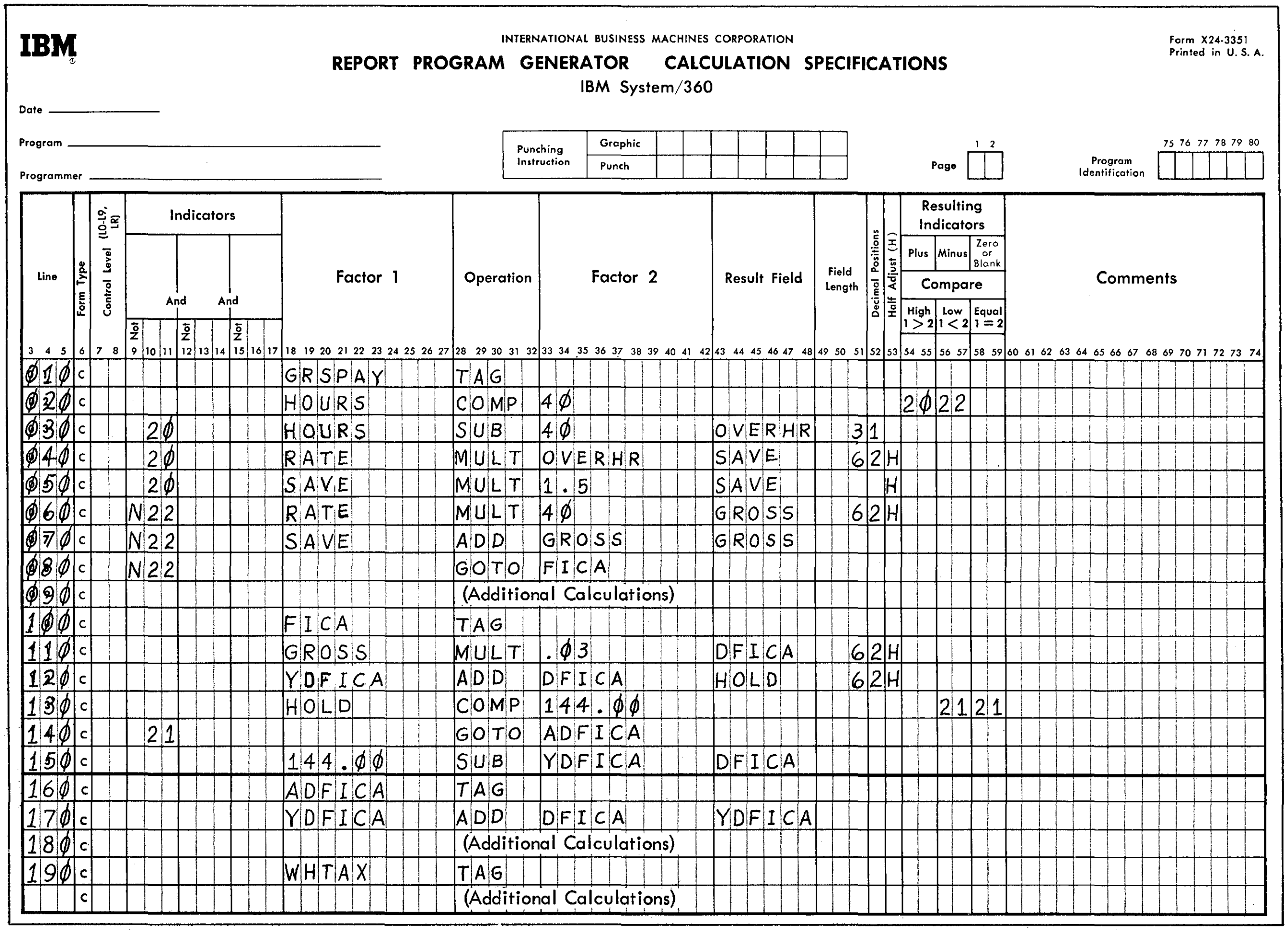
Figure 53 illustrates entries on the Calculation Specification sheet that are used in part of a payroll application. The entries on the sheet are discussed by line number.

Note: The blank spaces signify that additional calculations have been specified, but in this example they have been omitted.

Explanation

010 The program branches to GRSPAY from some other detail calculation (not shown in this example).

020 The number of hours the employee worked are compared with the literal 40. If the employee worked more than 40 hours, indicator 20 is turned on.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction | Graphic | Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR) (7-8)	Indicators (9-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators: Plus / Compare High 1 > 2 (54-55)	Minus / Low 1 < 2 (56-57)	Zero or Blank / Equal 1 = 2 (58-59)	Comments (60-74)
Ø1Ø	C			GRSPAY	TAG									
Ø2Ø	C			HOURS	COMP	4Ø					2Ø	22		
Ø3Ø	C		2Ø	HOURS	SUB	4Ø	OVERHR	3	1					
Ø4Ø	C		2Ø	RATE	MULT	OVERHR	SAVE	6	2	H				
Ø5Ø	C		2Ø	SAVE	MULT	1.5	SAVE			H				
Ø6Ø	C		N22	RATE	MULT	4Ø	GROSS	6	2	H				
Ø7Ø	C		N22	SAVE	ADD	GROSS	GROSS							
Ø8Ø	C		N22		GOTO	FICA								
Ø9Ø	C				(Additional Calculations)									
1ØØ	C			FICA	TAG									
11Ø	C			GROSS	MULT	.Ø3	DFICA	6	2	H				
12Ø	C			YDFICA	ADD	DFICA	HOLD	6	2	H				
13Ø	C			HOLD	COMP	144.ØØ						21	21	
14Ø	C		21		GOTO	ADFICA								
15Ø	C			144.ØØ	SUB	YDFICA	DFICA							
16Ø	C			ADFICA	TAG									
17Ø	C			YDFICA	ADD	DFICA	YDFICA							
18Ø	C				(Additional Calculations)									
19Ø	C			WHTAX	TAG									
	C				(Additional Calculations)									

Figure 53. Using the Calculation Specification Sheet

030-050 If indicator 20 is on, three calculations are performed. The literal 40 is subtracted from the number of hours worked and the overtime hours are placed in the field OVERHR, which is a three-position field with one decimal position. RATE is multiplied by OVERHR and the result is placed in the field SAVE, which is a six-position field with two decimal positions. SAVE is multiplied by the literal 1.5 (which is the overtime premium rate). The result is placed back in SAVE, and it is half-adjusted.

060 RATE is multiplied by 40, and the result is stored in GROSS. This operation is performed whether or not the employee worked any overtime. It is not performed if the employee has worked less than 40 hours.

070 GROSS is added to SAVE.

080 The program branches to the label FICA, if indicator 22 is off.

090 Additional calculations.

100 The operation code TAG provides the label FICA. The RPG program branches to this label.

110 GROSS is multiplied by the literal .03 and the result is placed in the field DFICA, which is a six-position field with two decimal positions. The result is half-adjusted.

120 DFICA is added to YDFICA and the result is placed in the field HOLD.

130 The contents of HOLD are compared with the literal 144.00, and if HOLD is less than, or equal to, 144.00, indicator 21 is turned on.

140 If indicator 21 is on, the program branches to the label ADFICA.

150 YDFICA is subtracted from the literal 144.00 and the result is placed in DFICA.

160 The operation TAG provides the label ADFICA to which the program can branch (either from the specification on line 140 or sequentially from the specification on line 180).

170 DFICA is added to YDFICA and the result is stored in YDFICA.

180 Additional calculations.

190 The operation code TAG provides the label WHTAX to which the RPG program can branch.

Entries on the Output-Format Specification sheet specify:

1. The kind of output that will be produced.

2. The location of the specific data fields in the output reports and records.

The specifications for this sheet can be divided into two categories as shown in Figure 54: file identification and control, and field description.

FILE IDENTIFICATION: These entries specify the output files and the records to be written on the file.

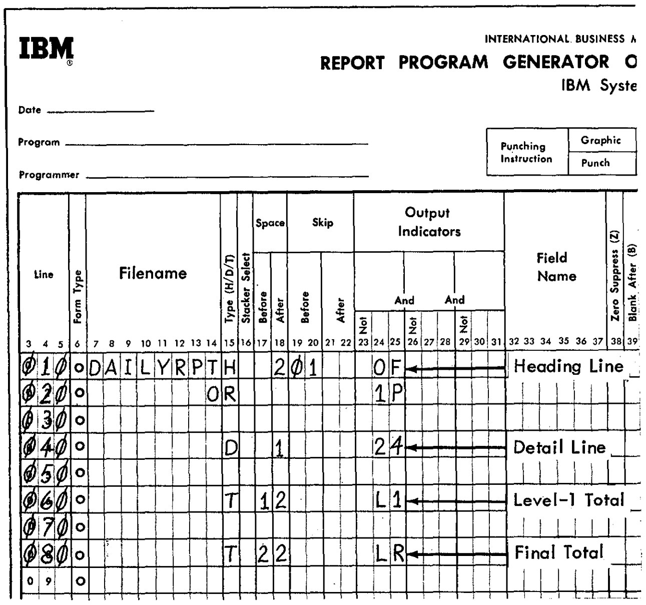
FIELD DESCRIPTION: These entries indicate the position of each field in the output record. Each field is written on a separate line of each sheet, and each field is written below its corresponding record entry.

FILE IDENTIFICATION AND CONTROL

FILE NAME (COLUMNS 7-14)

A file name is assigned to each output file. The file name must be left-justified, and it must begin with an alphabetic character. The file name may be alphameric, but it must not contain special characters or blanks.

When writing the output specifications, the file name need only be entered once. Enter it on the first line to define the file as shown in Figure 55.



IBM

INTERNATIONAL BUSINESS
REPORT PROGRAM GENERATOR O
IBM Syste

Date

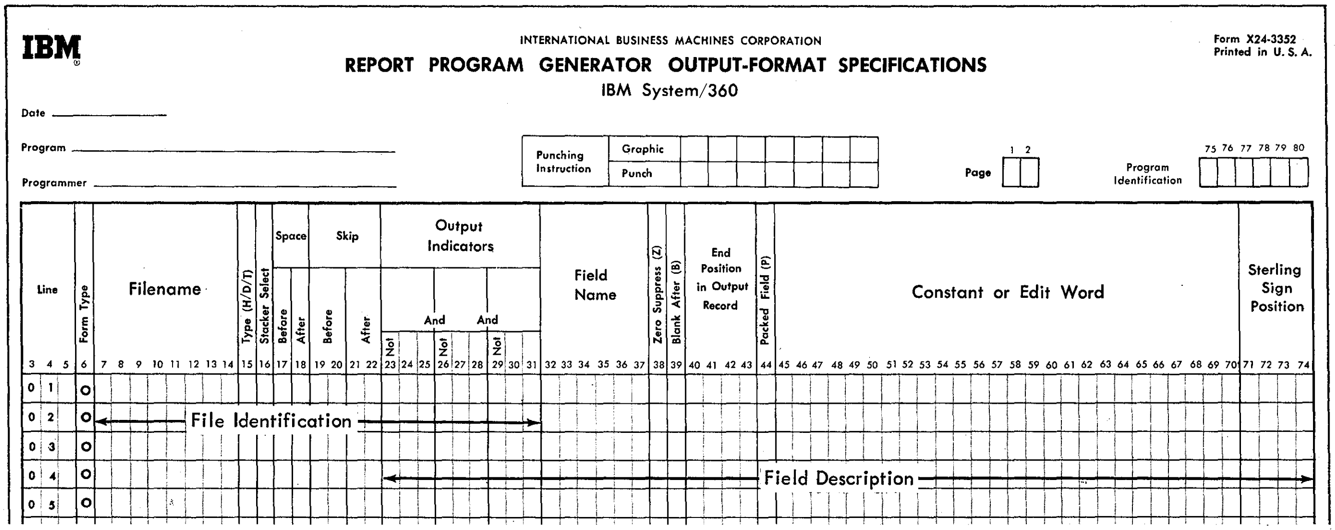
Program

Programmer

Punching Instruction	Graphic
	Punch

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators (23-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)
∅1∅	o	DAILYRPT	H			2	∅1		0F	Heading Line		
∅2∅	o	O	R						1P			
∅3∅	o											
∅4∅	o		D			1			24	Detail Line		
∅5∅	o											
∅6∅	o		T		1	2			L1	Level-1 Total		
∅7∅	o											
∅8∅	o		T		2	2			LR	Final Total		
09	o											

Figure 55. Specifying Heading, Detail, and Total Lines



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS
IBM System/360

Form X24-3352
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction	Graphic
	Punch

Page (1 2)

Program Identification (75 76 77 78 79 80)

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators: Not, And Not, And Not (23-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)	End Position in Output Record (40-43)	Packed Field (P) (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
01	O															
02	O	← File Identification →														
03	O															
04	O								← Field Description →							
05	O															

Figure 54. Output-Format Specification Sheet

TYPE H/D/T (COLUMN 15)

The entry in this column identifies the type of record being specified. The following three entries are used for this specification:

H	Heading Record
D	Detail Record
T	Total Record

All heading records for the file must be entered first, followed by all detail records, and then by all total records (see Figure 55, column 15).

HEADING RECORDS: These records usually contain constant information. However, they also contain information from input records, including the record present at the time the output record is produced.

DETAIL RECORDS: These records have a direct relationship to the input record. Most data in a detail record comes from the input record or from calculations performed at detail time.

TOTAL RECORDS: Operations upon fields from the input record are preceded by the test for control-field changes, the performance of total-time calculations, and the formation of total records. Thus, data from an input record that causes a control-field change cannot contribute data to total records that result from that control change. But heading and detail records can contain data from the input record.

STACKER SELECT (COLUMN 16)

If punched output occurs in the object program, a stacker number is entered in this column. The cards fall in the stacker that is indicated by this column. The stacker pockets and their acceptable codes are listed below:

IBM 1442

Pocket	Code
1	1 or blank
2	2

IBM 2520

Pocket	Code
1	1 or blank
2	2

IBM 2540

Pocket	Code
P1	1
P2	4 or blank
RP3	8

Note: When punched output is specified, the last card will not be put out automatically.

For information concerning the stacker selection of combined files, refer to the section Combined Files Stacker Select.

SPACE (COLUMNS 17-18)

This specification (and the next specification Skip, columns 19-22) is used to provide the proper spacing of printed reports.

Note: If the record is to be printed, at least one entry is required in columns 17-22.

Space Before (Column 17)

Enter in this column the number of lines to be spaced before the line is printed. Specify zero, one, two, or three spaces before printing by placing the entry 0, 1, 2, or 3 in column 17.

Space After (Column 18)

Specify zero, one, two, or three spaces after printing by placing the entry 0, 1, 2, or 3 in column 18. Enter a zero in this column to specify no space after printing.

SKIP (COLUMNS 19-22)

This specification provides for the proper spacing of reports. It is directly related to the function of the printer carriage-control tape.

Skip Before (Columns 19-20). The entries 01-12 cause the printer carriage to skip to channels 1 through 12 of the carriage tape before the line is printed.

Skip After (Columns 21-22). The entries 01-12 cause the printer carriage to skip to channels 1 through 12 of the carriage tape after the line is printed. In Figure 55, before the heading line is written, the form skips to channel 1.

> Note: The order in which spacing and skipping is performed is as follows:
>
> Skip Before
>
> Space Before
>
> Skip After
>
> Space After
>
> If both Space After and Skip After are specified, only Skip After is performed.

Overflow Indicator

Carriage overflow, causes the setting of the indicator as specified by the user in the file-description specifications (Columns 33-34).

When a punch in channel 12 of the control tape is sensed the overflow indicator is turned on. It remains on for one complete processing cycle; it turns off after the heading and detail lines of the next record are printed. Because the overflow indicator is on during calculations, it can control calculation specifications.

A test for the overflow status is made by the object program immediately before each line of the report is printed (but after any Space Before specifications are executed). Two conditions can occur at this time:

1. If the overflow indicator is on before a detail line is printed, the detail line, and any other detail lines whose output indicators are on, are printed. Any Space After specifications are executed, the next record is read, and a test is made to determine if a control break has occurred. If one has occurred, all total lines (caused by the control break) are printed, and then any overflow printing specified is performed.

2. If the overflow indicator is on before a total line is printed, all total lines, whose output indicators are on, are printed, and this is followed by any specified overflow printing.

Automatic Skipping

If an overflow indicator is not specified on the File Description Specification sheet for a file (and the Skip or Space columns on the Output-Format Specification sheet contain an entry) the RPG program provides automatic skipping from channel 12 to channel 1 for the object program.

If an overflow indicator is specified on the File Description Specification sheet for a file, and none of the records of the file are conditioned by overflow on the Output-Format Specification sheet, automatic skipping will also occur.

Printing Lines Conditioned by Overflow

In the upper half of Figure 56, the detail line 010 specifies that if a level-1 control break does not occur on this cycle, but has occurred on the previous cycle, indicator 25 is turned OFF. The total line 020 specifies that if a level-1 control break has occurred, indicator 25 is turned ON. Indicator 25 remains on for an extra total cycle because:

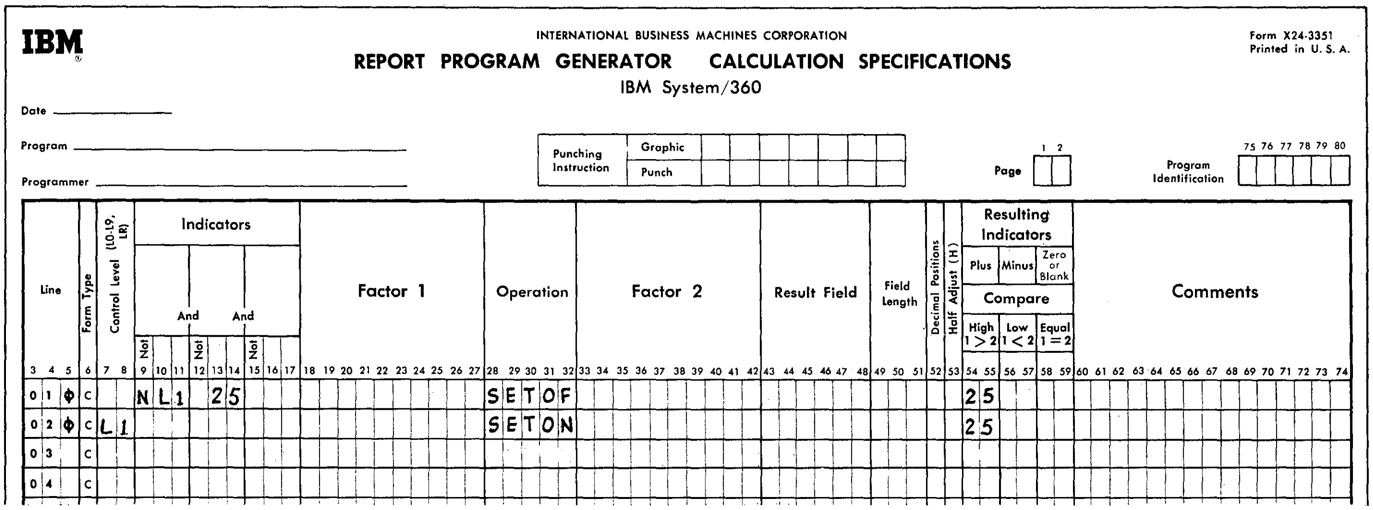
1. If overflow has occurred at detail time, it will be sensed at total time, and

2. The overflow routine is executed after the control level-1 indicator (L1) has been turned off.

To prevent the line from printing twice, thus providing an error in the printed report, the line is coded as shown in the lower half of Figure 56. By making the condition mutually exclusive, the line is not printed twice.

The order in which the object program prints lines conditioned by overflow follows:

1. When overflow occurs during the printing of a heading or detail line, the



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

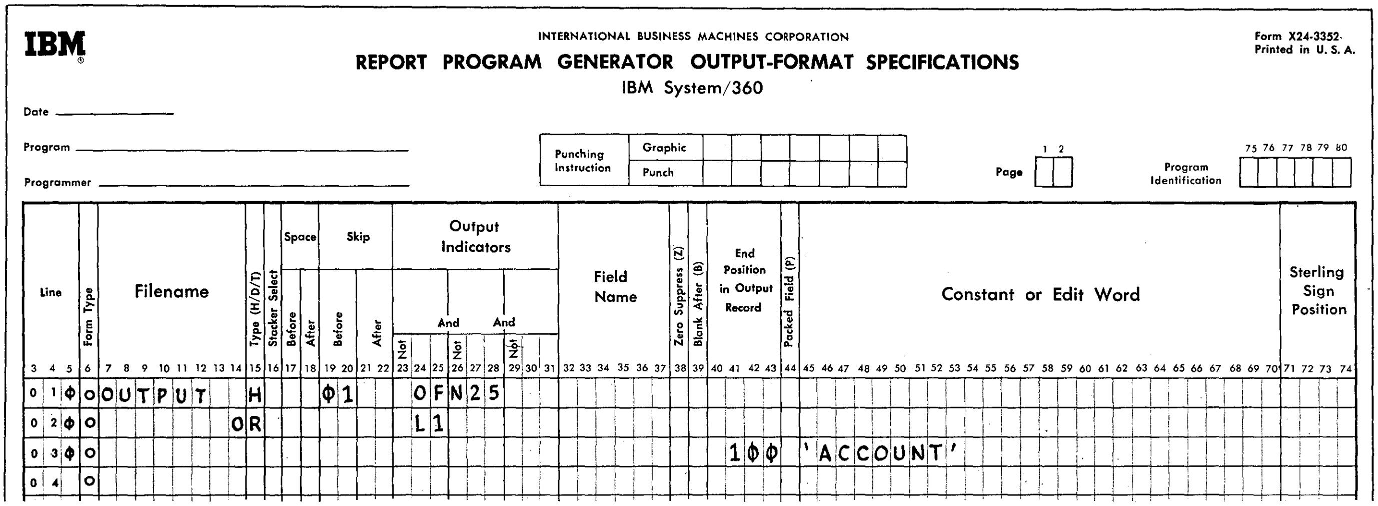
Date
Program
Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR) (7-8)	Indicators: Not (9)	(10-11)	And: Not (12)	(13-14)	And: Not (15)	(16-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators: Plus / Compare High 1>2 (54-55)	Minus / Low 1<2 (56-57)	Zero or Blank / Equal 1=2 (58-59)	Comments (60-74)
01Φ	C		N	L1		25				SETOF						25			
02Φ	C	L1								SETON						25			
03	C																		
04	C																		



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space: Before (17)	Space: After (18)	Skip: Before (19-20)	Skip: After (21-22)	Output Indicators: Not (23)	(24-25)	And: Not (26)	(27-28)	And: Not (29)	(30-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)	End Position in Output Record (40-43)	Packed Field (P) (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
01Φ	O	OUTPUT	H				Φ1			OF	N	25									
02Φ	O	OR								L1											
03Φ	O																	10Φ		'ACCOUNT'	
04	O																				

Figure 56. Specifying Output Indicators

object program does not immediately print lines conditioned by overflow. The program does, however, take note that the overflow condition has occurred.

2. All total calculations are executed at this time, including the total calculations conditioned on overflow.

3. Total lines not conditioned by overflow are printed.

4. The object program then tests to see if overflow has occurred.

5. Total lines conditioned by overflow are printed.

6. Heading and detail lines conditioned by overflow are printed.

7. All detail calculations are executed at this time, including the detail calculations conditioned on overflow (see Figure 26).

Note: If an overflow condition has been specified, automatic skipping is not performed.

Multiple Printers

With this program, it is possible to use a maximum of three output printer files for each RPG object program. When producing records for more than one printer, the user specifies a unique overflow indicator for each printer.

In columns 33-34 of the File Description Specification sheet, one of three available overflow indicators is entered. The line on the File Description Specification sheet which is used to define the output printer file may contain one of these indicators in columns 33-34: OA, OF, or OV.

OUTPUT INDICATORS (COLUMNS 23-31)

This specification may be used for either file identification (in this case, at least one output indicator must be specified) or field description. Entries in this column may specify a maximum of three indicators. These indicators control:

1. When the line is to become output, or

2. When a particular field is to be written.

The following information applies only to its use in the file identification line.

If more than one indicator is specified on one line, all indicators are considered in and AND relationship. That is, all conditions specified must be satisfied before the output condition can be executed.

If the object program requires that more than three indicators be in and AND relationship, the letters AND are entered in columns 14-16 of the following line, and the additional indicators are specified on that line. If one of the indicators is an overflow indicator (OA, OF, or OV), it must not appear on the same line as the AND.

If the output condition is executed in an OR-relationship (one or the other of two sets of conditions) the letters OR are entered in columns 14-15 of the following specification line, and the OR indicators are specified on that line.

Additional specification lines can specify as many output indicators in either an AND or OR relationship as required by the object program. Each additional line must begin with AND or OR in column 14.

Permitted Entries for Output Indicators, Columns 24-25, 27-28, and/or 30-31

Enter in these columns:

1. Resulting indicators (Columns 19-20 of the Input Specification sheet or columns 54-59 of the Calculation Specification sheet).

2. Special indicators (LR, H1, MR, etc).

3. Field indicators (columns 65-70 of the Input Specification sheet).

   Note: When an output record is not conditioned by any resulting indicators, the record is conditioned 00.

NOT (COLUMNS 23, 26, AND/OR 29)

If any of the output indicators must be off, enter an N in column 23, 26, or 29 (whichever is appropriate).

EXAMPLES OF OUTPUT INDICATORS

Figure 57 shows six examples of output indicators used with the output file. The numbers to the right refer to the following discussion:

1. The control carriage is skipped to channel 02 before printing, and the heading line is printed only when an overflow condition occurs or when the 1P (first page) indicator is on.

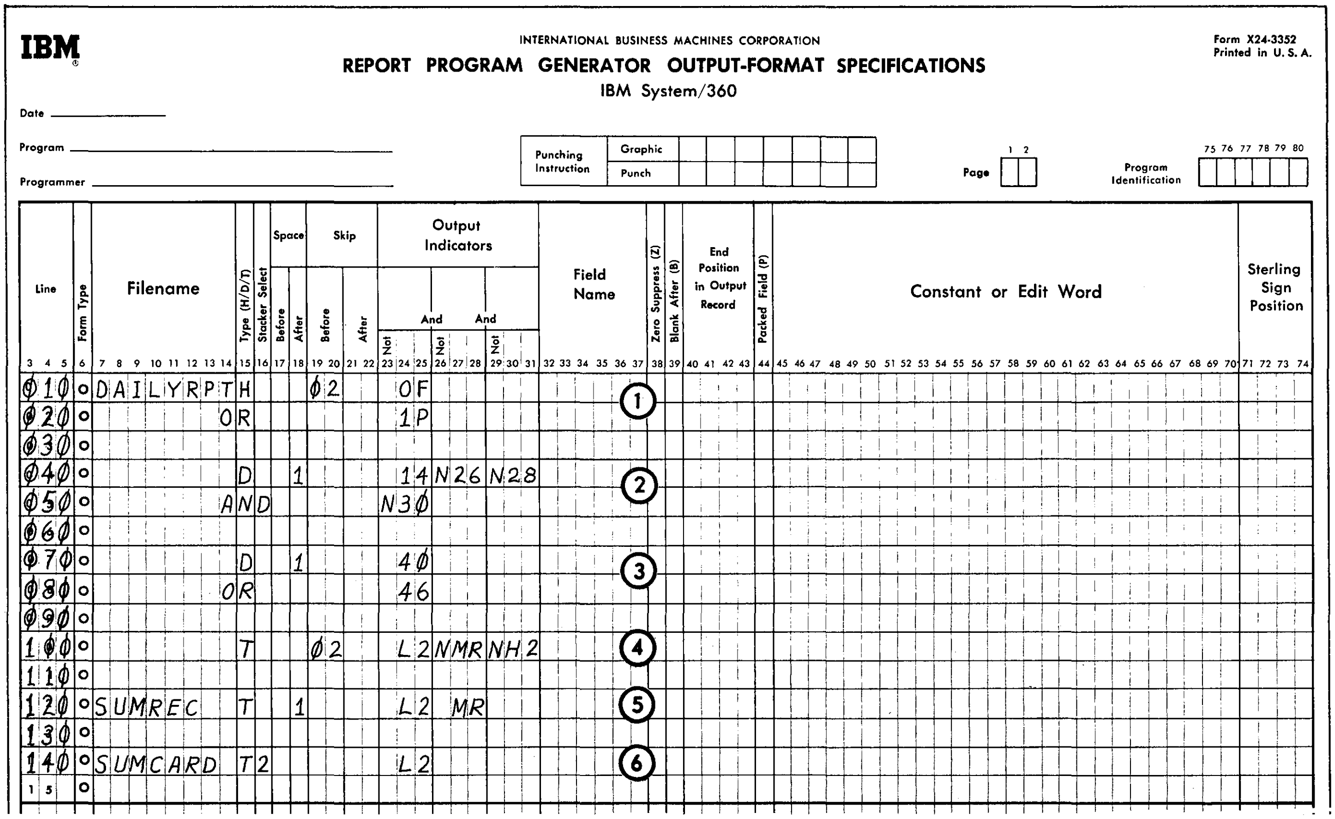
2. The detail line is printed only if indicator 14 is on, and indicators 26, 28, and 30 are off. Indicators 26, 28, and 30 could be field indicators from the input specifications, or they could be resulting indicators from the calculation specifications.

3. The detail line is printed if indicator 40 or 46 is on.

4. The total line is printed and the control carriage is skipped to channel 2 before printing only if the level-2 indicator is on, and the MR indicator is off, and H2 is off. The specification NH2 prevents the object program from printing a line if an error condition has occurred. The program does not stop until after all processing for the record has been completed. This feature enables the programmer to prevent the printing of erroneous data.

5. The summary record is printed at the level-2 control time, but the MR (matching records) indicator must also be on.

6. A summary card punched at the level-2 control time is selected into stacker 2 of an IBM 1442 Card Read-Punch.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page (1 2)

Program Identification (75 76 77 78 79 80)

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators Not (23)	Output Indicators (24-25)	And Not (26)	And (27-28)	And Not (29)	And (30-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)	End Position in Output Record (40-43)	Packed Field (P) (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
φ1φ	O	DAILYRPT	H				φ2			OF					①						
φ2φ	O	OR	R							1P											
φ3φ	O																				
φ4φ	O		D			1				14	N	26	N	28	②						
φ5φ	O	AND	D						N	3φ											
φ6φ	O																				
φ7φ	O		D			1				4φ					③						
φ8φ	O	OR	R							46											
φ9φ	O																				
1φφ	O		T				φ2			L2	N	MR	N	H2	④						
11φ	O																				
12φ	O	SUMREC	T			1				L2		MR			⑤						
13φ	O																				
14φ	O	SUMCARD	T	2						L2					⑥						
15	O																				

Figure 57. Specifying Output Indicators

This concludes the description of the file-identification specifications. The remaining descriptions of the output form are concerned with the field-description section.

FIELD DESCRIPTION

These entries include specifications about:

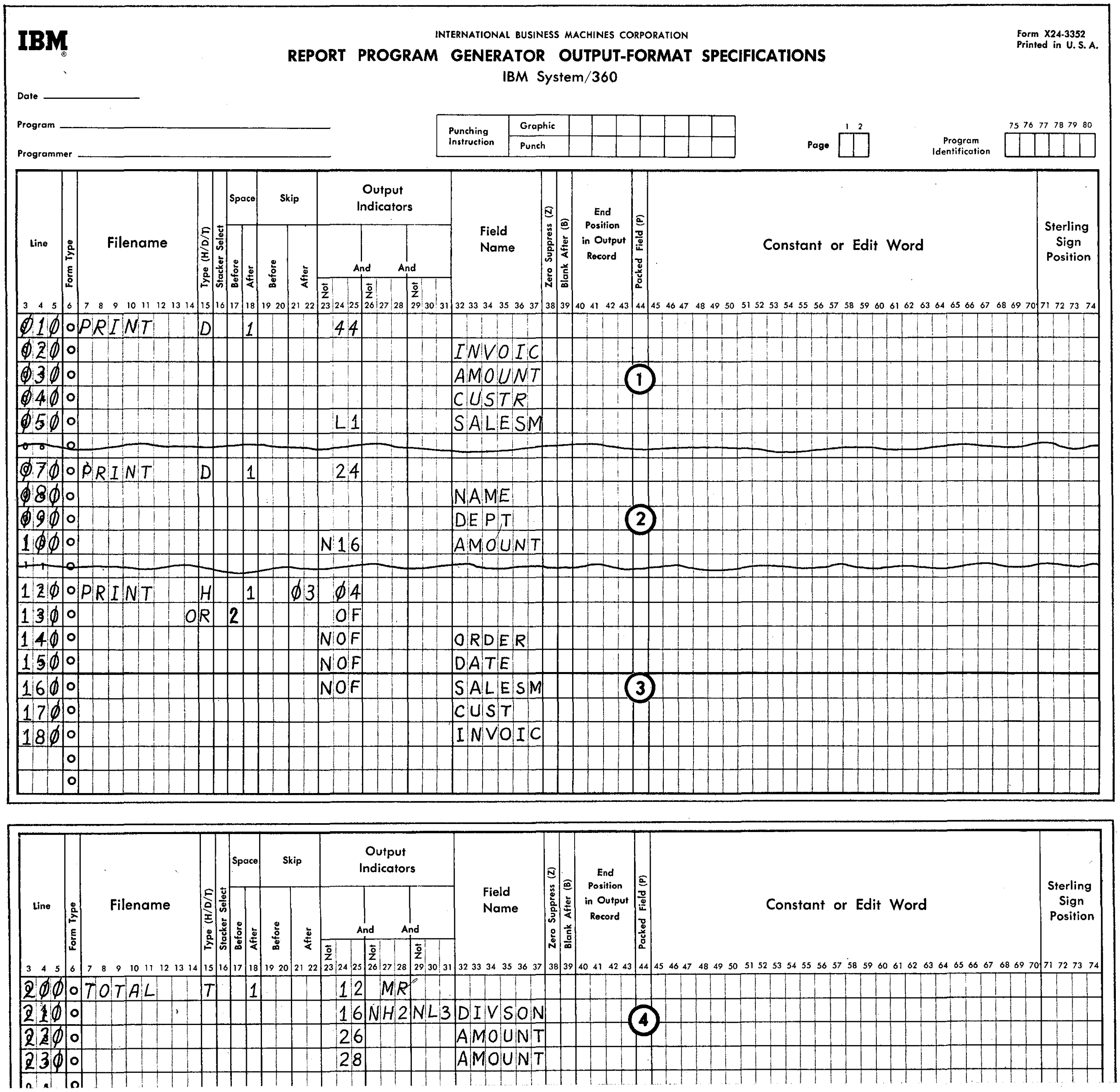
1. The control of the individual fields of a record, and

2. The output format of individual fields of a record. The fields of the record are written on the lines <u>below</u> their corresponding file entries. Each field is described on a separate line, and no entries are permitted in columns 7-22 of a field-description line.

OUTPUT INDICATORS (COLUMNS 23-31)

The same types of indicators used for file identification can be used for field description. The maximum number of indicators that can be considered in an AND relationship is three; all must be specified on one field-description line.

Figure 58 shows four sets of indicators used as output indicators for field-description lines. The numbers to the right of the figure correspond to the following list.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page

Program Identification

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators	And	And	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
Ø1Ø	O	PRINT	D			1			44									
Ø2Ø	O											INVOIC						
Ø3Ø	O											AMOUNT						
Ø4Ø	O											CUSTR						
Ø5Ø	O								L1			SALESM						
Ø7Ø	O	PRINT	D			1			24									
Ø8Ø	O											NAME						
Ø9Ø	O											DEPT						
1ØØ	O								N16			AMOUNT						
12Ø	O	PRINT	H			1		Ø3	Ø4									
13Ø	O		OR		2				OF									
14Ø	O								NOF			ORDER						
15Ø	O								NOF			DATE						
16Ø	O								NOF			SALESM						
17Ø	O											CUST						
18Ø	O											INVOIC						
	O																	
	O																	

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators	And	And	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
2ØØ	O	TOTAL	T			1			12	MR								
21Ø	O								16	NH2	NL3	DIVSON						
22Ø	O								26			AMOUNT						
23Ø	O								28			AMOUNT						

Figure 58. Specifying Field Names

1. Four fields are printed from a detail record identified by indicator 44: invoice, amount, customer, and salesman. The entry L1 causes the field, SALESM, to be printed only for the first detail record of each control group. (Remember that a control level indicator remains on during the following detail calculation and print cycle.)

   In this example, the salesman field is group indicated.

2. The second example illustrates how to prevent the printing of just one field of a record. The field AMOUNT is printed only if indicator 16 is off. This indicator might be a resulting indicator used to determine if the calculated field AMOUNT is zero.

3. This example selectively prints the field headings for an invoice form. This specification prints all the heading information on the first form, but if the information for one customer

order continues on two or more forms, only the customer and invoice fields on succeeding forms are printed.

Printing of the entire line is controlled by indicators 04 or OF. In the field-description specifications, the OF indicator is also used to prevent printing of fields ORDER, DATE, and SALESM when an overflow condition occurs.

In column 17 of line 130, the 2 causes selective spacing of the OR'ed line. If 04 is on, a skip after to channel 3 will be made with no spaces before. If OF is on, 2 spaces before and no skip after will be made. If columns 17-22 of the line were blank, the space/skip specifications from line 120 would apply.

4. The last example illustrates how a field can be controlled for printing by an AND relationship and an OR relationship.

   The field DIVSON is controlled by three AND indicators: 16, NH2, and NL3. The field AMOUNT is controlled by two OR conditions. In the field description line, the OR relationship is used by writing the field name twice and specifying each appropriate OR indicator. The letters OR cannot be specified in columns 14-15 of a field-description line.

FIELD NAME (COLUMNS 32-37)

This specification identifies each field of the record to be written. The fields may be listed on the form in any sequence. The order in which they appear in the output record is determined by the entry in columns 40-43.

Enter in columns 32-37 the name of the field that is defined on the line. The field name must have been previously defined on either the input or calculation sheet. If a constant is to be written, it is specified in Constant or Edit Word (columns 45-70), and Field Name is left blank.

If the value of the field is numeric, the sign of the field will appear in the units position unless the field is edited, zero suppressed, or the zone of the field has been blanked out on the Calculation Specification sheet by a numeric literal.

Examples of field names are given in Figure 58.

ZERO SUPPRESS (COLUMN 38)

An entry of Z in this column causes zero-suppression of the field in the output record. Zero-suppression means that the zeros to the left of the significant digits in the field are not written on the output file. Fields to be zero suppressed must be specified as numeric.

An entry of Z in this column also provides another function. Zone bits in the units position of a field are removed in the output record before the field is written.

If an edit-control word is specified for the field, this column must be blank.

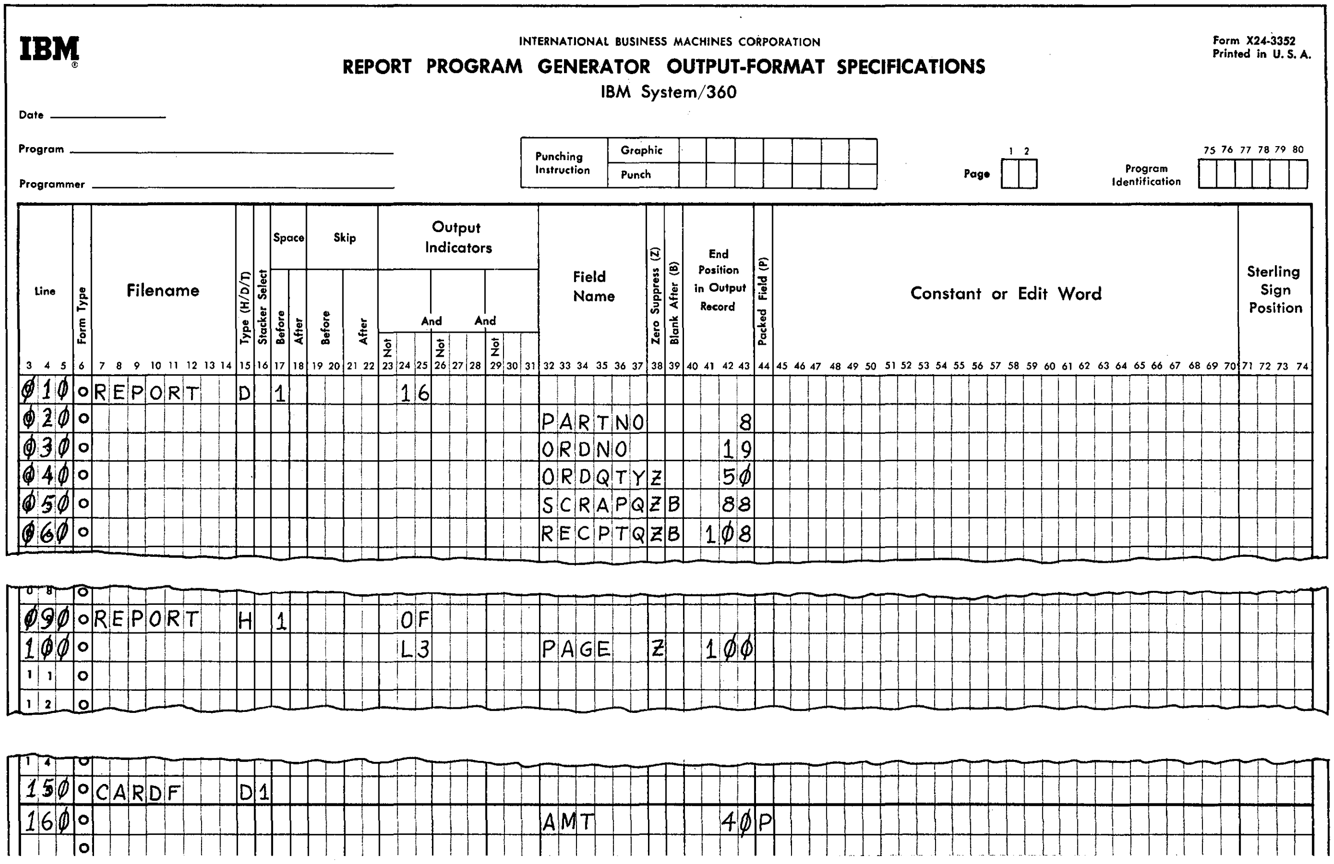
BLANK AFTER (COLUMN 39)

If a B is entered in this column, the output field will be reset to blanks or zeros after the field is moved to the output area. Alphameric fields are set to blanks, and numeric fields are set to zeros. If the field name specified is a table name, only the table hold area is set to blanks.

This specification has an additional feature. It allows the programmer to provide for resetting field indicators or resulting indicators that test for blanks or zero. If the output field being reset by the blank-after specification, is also being tested for zero or blank on the input specifications, or for zero on the calculation specifications, the corresponding indicator is turned on after the output field is reset. If multiple Zero or Blank indicators are specified for the same field name, only the last indicator specified will be turned on by the Blank After specification.

END POSITION IN OUTPUT RECORD (COLUMNS 40-43)

This specification indicates the exact location of the field in the output record. In columns 40-43, enter only the position in the output record where the rightmost (low-order) character of the field is to be located.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date ____________

Program ____________

Programmer ____________

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators Not (23)	(24-25)	And Not (26)	(27-28)	And Not (29)	(30-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)	End Position in Output Record (40-43)	Packed Field (P) (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
Ø1Ø	o	REPORT	D		1					16											
Ø2Ø	o														PARTNO			8			
Ø3Ø	o														ORDNO			19			
Ø4Ø	o														ORDQTY	Z		5Ø			
Ø5Ø	o														SCRAPQ	Z	B	88			
Ø6Ø	o														RECPTQ	Z	B	1Ø8			
0 8	o																				
Ø9Ø	o	REPORT	H		1					OF											
1ØØ	o									L3					PAGE	Z		1ØØ			
1 1	o																				
1 2	o																				
1 4	o																				
15Ø	o	CARDF	D	1																	
16Ø	o														AMT			4Ø	P		
	o																				

Figure 59. Positions in Output Record

Assume that a ten-position amount field is to be printed in print positions 21 through 30. The entry in columns 42-43 would be 30. Columns 40-41 are left blank, because the leading zeros may be omitted.

PACKED OUTPUT (COLUMN 44)

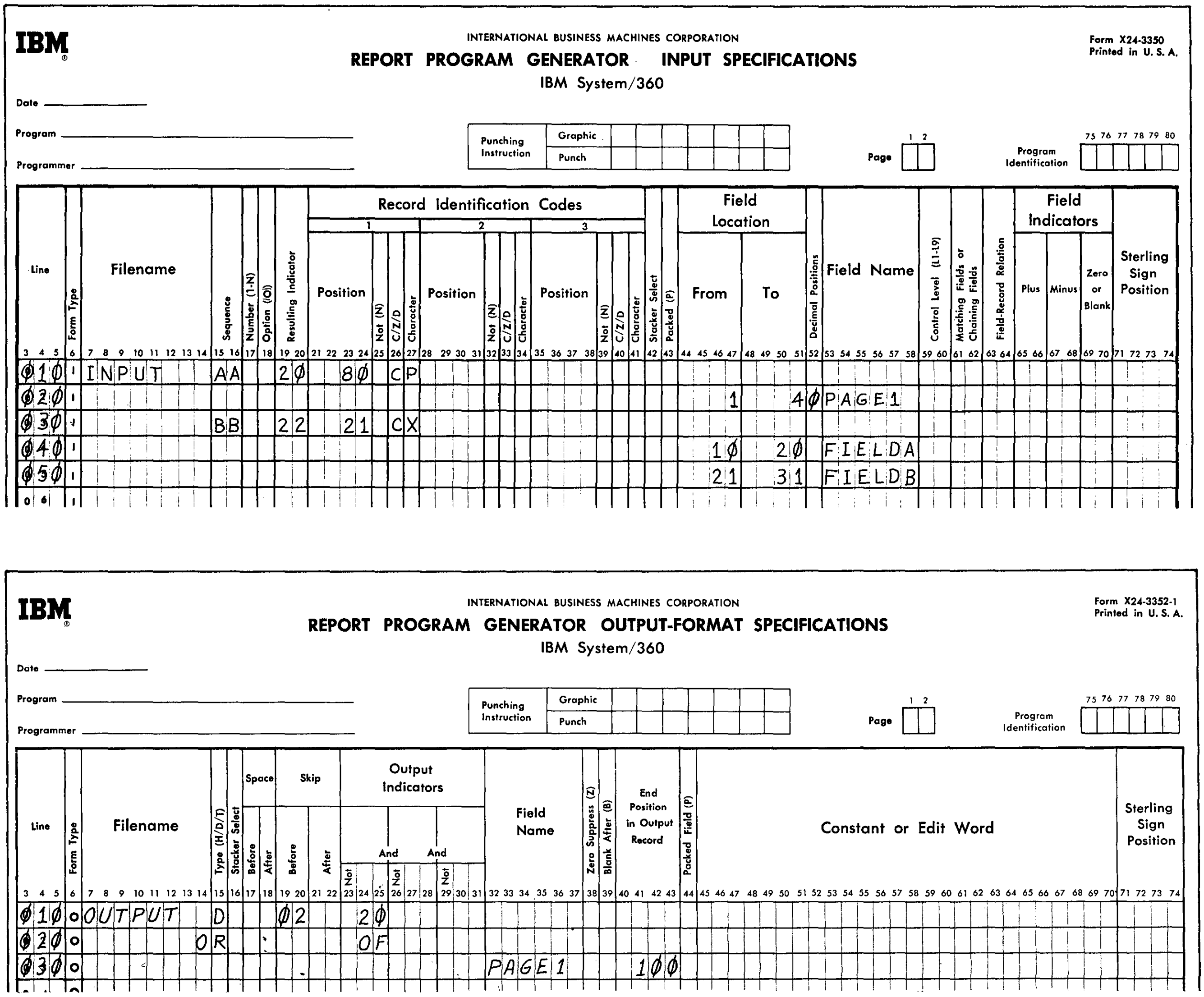
Enter a P in this column if the output field is to appear in the packed decimal format. Otherwise, leave this column blank. If the output is to be printed, leave this column blank.

Page Numbering

Page numbering is another feature of RPG. By placing the word PAGE in columns 32-37 (Field Name) of the output specifications, automatic page numbering will be obtained. The page entry in Field Name of Figure 59 causes each page of the output to be numbered consecutively in print-positions 97 to 100. The number is always four positions long and must be specified as numeric. However, zero suppression can be performed. The page number is increased by one before it is printed.

Page numbering normally begins with the number 1, but page numbering can be started with any number by preparing a record-type that contains the starting page, less 1.

In this case the record must be defined on the Input Specification sheet with a field labeled PAGE (n). (The (n) following page is explained in the section Page Numbering for Multiple Printers.) Figure 60 shows an example of this specification. In this example the page is initialized from sequence AA of the input sheet. It must be numeric (entry in column 52 of the Input Specification sheet). On the output sheet it is defined as a separate field that will print in position 100. If the page numbering were to begin with the number 500, 499 would be punched in columns 1-4 of the input record that contains the character P in column 80.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Line	Form Type	Filename	Sequence	Resulting Indicator	Record Identification Codes 1 Position	C/Z/D	Character	Field Location From	To	Decimal Positions	Field Name
Ø1Ø	I	INPUT	AA	2Ø	8Ø	C	P				
Ø2Ø	I							1	4	Ø	PAGE1
Ø3Ø	I		BB	22	21	C	X				
Ø4Ø	I							1Ø	2Ø		FIELDA
Ø5Ø	I							21	31		FIELDB

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352-1
Printed in U. S. A.

Line	Form Type	Filename	Type (H/D/T)	Skip Before	Output Indicators	Field Name	End Position in Output Record
Ø1Ø	O	OUTPUT	D	Ø2	2Ø		
Ø2Ø	O	OR			OF		
Ø3Ø	O					PAGE1	1ØØ

Figure 60. Page Numbering

A page number can be reset and a new series of page numbers may be started during the processing of the object program. When PAGE appears in the field name of the Output-Format Specification sheets, the output indicators are used to reset the page to number 1. In this case, the indicator must be placed in <u>Output Indicators</u> on the same line as the field name specification PAGE (see the L3 indicator in Figure 59).

PAGE NUMBERING FOR MULTIPLE PRINTERS

The individual printer files can initialize page numbering. Special <u>Page</u> entries (PAGE, PAGE1, and PAGE2) can be initialized to any count in the same manner as described for <u>PAGE</u>.

CONSTANT OR EDIT WORD (COLUMNS 45-70)

Columns 45-70 provide the ability:

1. To include constants in output records, and
2. To permit editing of numeric fields.

<u>Constants</u>

A literal of up to twenty-four alphameric characters may be placed in columns 45-70. The literal must begin in column 45, and it must be enclosed by a set of single quotation signs even if it contains only numeric characters. The literal beginning in column 45 will be placed on the output record as defined in End Position in Output Record, columns 40-43.

Rules for Forming Alphameric Literals in an Output Record:

1. Any character in the character set may be used in an alphameric literal. Blanks are treated as valid characters.

2. Alphameric literals must be enclosed in a set of single-quote symbols. A single quote symbol may be contained within a literal by entering an additional single quote symbol within the literal. For example, the literal 5 o'clock would be entered as '5 o''clock' in columns 45-56 of the Output-Format Specification sheet.

3. Field Name (Columns 32-37) must be left blank when an alphameric literal is defined on the line.

Edit Words

An edit word provides for the punctuation of amount fields, including the printing of dollar signs, commas, periods, and sign status. An edit operation is shown in Figure 61.

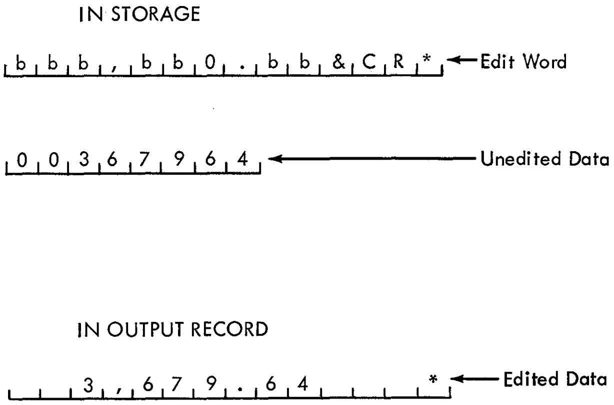


Figure 61. Edit Operation

When an amount field is to be edited, its edit word is placed in columns 45-70 of the line which specifies the field. An edit word consists of two parts (the body and the status) as shown in Figure 62.

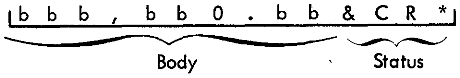


Figure 62. Two Parts of an Edit Word: Body and Status

The body of the edit word is the portion beginning with the leftmost blank, zero, or asterisk, and continuing to the right to the character that governs the transfer of the units position of the data field.

The status of the edit word is the portion continuing to the right from the body to the CR (credit), or - (minus) if present.

Rules for Forming an Edit Word:

1. An edit word must be enclosed in a set of single-quote symbols.

2. A blank in the edit word is replaced with the character from the corresponding position of the data field specified in Field Name.

3. An ampersand causes a space in the edited field.

4. A zero is used for zero-suppression. It is put in the rightmost position where zero suppression is to stop. It is replaced with the character from the corresponding position of the data field, unless that character is a zero. Column 38 (zero-suppression) must be left blank.

5. An asterisk in the body of the control word is used for asterisk protection and zero suppression. It is put in the rightmost position where zero suppression is to stop. It is replaced with the character from the corresponding position of the data field unless that character is a zero. Each zero that is suppressed is replaced by an asterisk. When an asterisk is encountered in an editword before a CR symbol, a minus symbol, or a zero, the asterisk is interpreted as representing asterisk protection.

6. A dollar sign in the body of the edit word written immediately to the left of the zero-suppression code (0) causes the insertion of a dollar sign in the position to the left of the first significant digit. This is called the floating dollar sign. A dollar sign that is entered immediately after the

initial single-quote mark will be fixed. That is, it will be printed in the same location each time. This is called the fixed dollar sign.

Note: When using the floating dollar sign, the edit word must contain the same number of blanks and/or zeros as the field to be edited contains digit positions.

7. The decimal and commas are printed in the edited output field in the same relative positions they were written in the edit word unless they are to the left of significant digits. In that case, they are blanked out or replaced by an asterisk.

8. All other characters used in the body of the edit word are printed if they are to the right of significant digits in the data field. If they are to the left of high-order significant digits in the data word, they are blanked out, or if asterisk protection is used, they are replaced by an asterisk.

9. The letters CR or the minus symbol in the status portion of the edit word are undisturbed if the sign in the data field is minus. If the sign is plus, or if the value of the field is minus zero, they are blanked out.

10. Asterisks to the right of the status portion of the edit word are undisturbed. They are normally used to indicate a specific class of total.

Figure 63 illustrates the use of constants and edit words. The numbers to the left correspond to the following list:

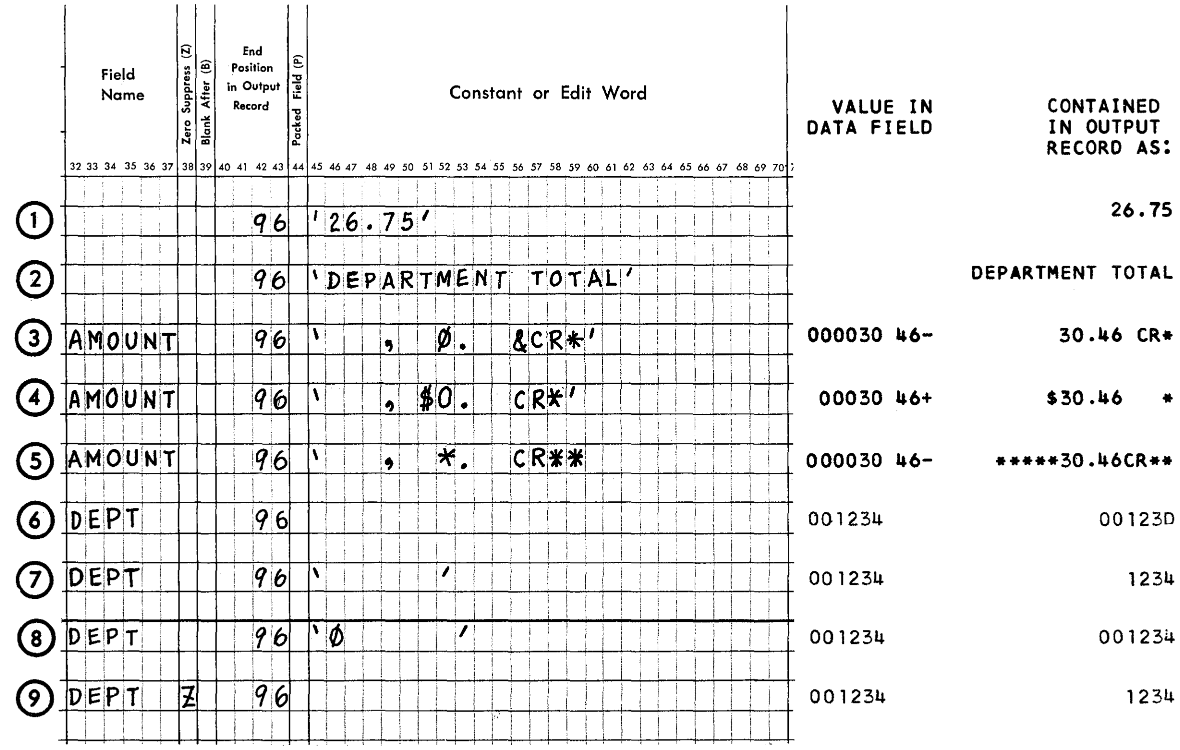
1. The constant 26.75 is in the output record ending in position 96. The field-name specification must be blank.

2. The constant DEPARTMENT TOTAL is contained in the output record ending in position 96. The field name must be blank.

3. This example illustrates zero-suppression to the left of significant digits. The letters CR are written because the amount field might be minus.

4. In this example, the floating dollar-sign protection enters the $ to the left of the first significant digit.

5. Asterisk protection enters as many asterisks to the left of the first significant digit as required to fill out the number of positions specified in the edit word.



	Field Name 32 33 34 35 36 37	Zero Suppress (Z) 38	Blank After (B) 39	End Position in Output Record 40 41 42 43	Packed Field (P) 44	Constant or Edit Word 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	VALUE IN DATA FIELD	CONTAINED IN OUTPUT RECORD AS:
1				96		'26.75'		26.75
2				96		'DEPARTMENT TOTAL'		DEPARTMENT TOTAL
3	AMOUNT			96		' , ∅. &CR*'	000030 46-	30.46 CR*
4	AMOUNT			96		' , $0. CR*'	00030 46+	$30.46 *
5	AMOUNT			96		' , *. CR**	000030 46-	*****30.46CR**
6	DEPT			96			001234	00123D
7	DEPT			96		' '	001234	1234
8	DEPT			96		'∅ '	001234	001234
9	DEPT	Z		96			001234	1234

Figure 63. Using Constants, Edit Words, and Zero Suppression

6. DEPT is a numeric field for which no editing or zero suppression is specified. The output field will contain a zone in the units position.

7. An edit word containing blanks only will suppress all high-order zeros in the output field.

8. The edit word has been specified as one digit longer than DEPT, with a zero in the high-order position. This allows all zeros in DEPT to print, but the zone of the units position has been stripped.

9. This line notes the use of zero suppression. The output field will contain no high-order zeros as in line 7.

STERLING (COLUMNS 71-74).

Enter in these columns the position in the record that will contain the sign of the sterling field. Leading zeros may be omitted. Enter an S in column 74 if the sign is in the normal position. If the Sterling specification is not required, leave this column blank. Additional information on Sterling is in Appendix C: Sterling Routines for the Report Program Generator.

This sheet provides information to RPG about:

1. The input files, defined on the Input Specification sheet, from which the object program will obtain data records. A maximum of three input files (including combined files) may be specified, (excluding RAF files, table files, and chaining files).

2. The input files used by the object program, such as record address files, and table files.

3. The output files are defined on the Output-Format Specification sheet on which the object program will write data records. Each file used by the object program will be defined. Each line of the File Description Specification sheet is used to define one file.

MAXIMUM NUMBER OF FILES AVAILABLE

The maximum number of files that can be used in the program is 10. The following list defines the maximum number of files for the various types of files that can be used. Any combination of these, up to 10, is permitted.

Type of File	Maximum Number
Primary	1
Secondary	2
RAF	1
Chained	3
Table	4
Output	
Printer	3
Other Output Units	9

FILE NAME (COLUMNS 7-14)

Each file used in the program is identified by writing the name of the file in columns 7-14. File names must be alphameric and left-justified (the file name must begin in column 7). The file name entered in these columns must also be entered on the Input Specification sheet (for input data files), on the Output-Format Specification sheet (for output data files), or on the File Extension Specification sheet.

> Note: The filename may be eight characters long but RPG will utilize only the first seven characters for all files except Indexed-Sequential files. For Indexed-Sequential files only the first five characters of the filename will be utilized in the RPG Program. Therefore, it is important that the first seven (or five) characters provide a unique filename.

FILE TYPE (COLUMN 15)

An entry in this column specifies the type of file defined on the line of the sheet. The following entries are allowed in this column:

Entry	Type of File	Explanation
I	Input	Identifies the file as an input file (it may be a record-address file, table file, or a file containing input data records).
O	Output	Identifies the file as an output file (it may be an updated table file to be written out). Column 39 must be left blank.
U	Update	Identifies the file as an update file (DASD only). An update file is both an input and an output file. The file is an update file if the object program alters the data in one or more fields of each record contained in the file without changing: 1. The nature of the data, 2. The length of the field in which the data is found, or 3. The location of the field.

A chained file may be updated at detail time or at total time. All other DASD files can only be updated at detail time.

C Combined Identifies the file as a combined file. A <u>combined file</u> is a card file containing cards read into the system and cards used for punching. It must reside on an IBM 1442 Card Read-Punch, or an IBM 2540 Card Read-Punch equipped with the Punch-Feed-Read Special Feature. Reading and then punching into the same file is accomplished in two different ways:

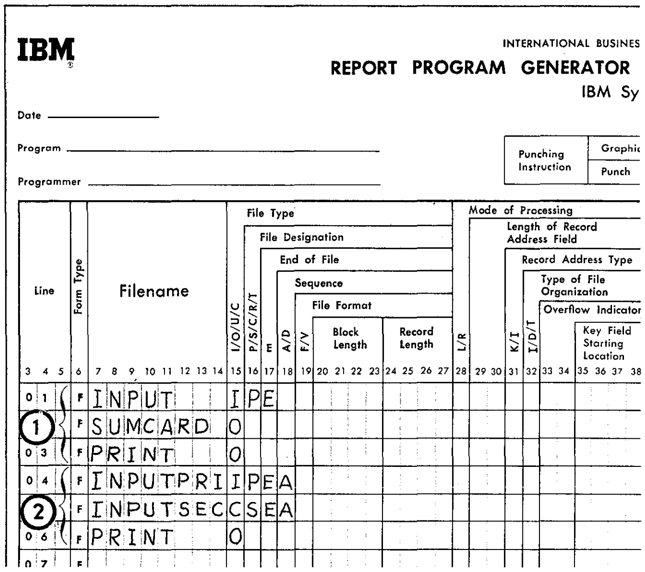
1. Punching into the same card that is read.
2. Punching into a blank trailer card in the same file.

A combined file differs from an update file because in the update file the input fields are revised or updated.

<u>Note</u>: RPG does not support the use of the IBM 2520 Card Read-Punch for combined files.

Figure 64 illustrates two examples of <u>File Name</u> and <u>File Type</u> specifications.

1. Detail cards are read into the system in one file, summary cards are punched into a separate file (which, in this example, might be the punch feed of an IBM 2540 and might contain blank cards only), and the printed report is in a third file.
2. There are two input files in this example. The second file (INPUTSEC) also contains cards that will be used for punching output data; therefore it is specified with file-type C.



IBM

INTERNATIONAL BUSINES

REPORT PROGRAM GENERATOR

IBM Sy

Date

Program

Programmer

Punching Instruction | Graphic | Punch

Line (3 4 5)	Form Type (6)	Filename (7–14)	File Type: I/O/U/C (15)	File Designation: P/S/C/R/T (16)	End of File: E (17)	Sequence: A/D (18)	File Format: F/V (19)	Block Length (20–23)	Record Length (24–27)	L/R (28)	Mode of Processing (29 30)	Length of Record Address Field / Record Address Type: K/I (31)	Type of File Organization: I/D/T (32)	Overflow Indicator (33 34)	Key Field Starting Location (35 36 37 38)
0 1	F	INPUT	I	P	E										
1	F	SUMCARD	O												
0 3	F	PRINT	O												
0 4	F	INPUTPRI	I	P	E	A									
2	F	INPUTSEC	C	S	E	A									
0 6	F	PRINT	O												
0 7	F														

Figure 64. Specifying File Name and File Type

FILE DESIGNATION (COLUMN 16)

Column 16 designates the type of input file being defined on this line of the File Description Specification sheet. The five entries permitted in this column are listed here. Detailed explanations of chained files, table files, record address files, primary files, and secondary files may be found in the sections <u>Using Tables and Exit Routines in the Object Program</u> and <u>Processing Multiple Input Files</u>.

If the file is an output file, leave this column blank. An entry must be made if the file is an input, update, or combined file. Enter in column 16 the following codes:

<u>Entry</u> <u>Explanation</u>

P The file defined is a primary file. Only one primary file may be defined.

S The file defined is a secondary file.

R The file defined is a Record Address File which relates to a direct access storage file.

C The file defined is a chained file (as defined under <u>Chaining</u>).

T The file defined is a table file.

END OF FILE (COLUMN 17)

Enter an E in column 17 if the input file is a sequential file and it is to be checked for an end-of-file condition. For one input file, the E entry will cause the LR (last record) indicator to turn on when the last record of the file has been processed.

For multiple input files, the end-of-job condition (LR) will occur when all the input files (defined on this sheet with an E in column 17) have reached the end-of-file.

If this column is left blank for all input files, the end-of-job condition occurs when all input files have been processed.

SEQUENCE (COLUMN 18)

This entry is normally made if there is more than one input file, and the matching-fields specification (columns 61-62 of the Input Specification sheet) is used. An entry in this column indicates whether the matching fields are in ascending or descending sequence.

Enter an A in column 18 if the matching fields are in ascending order, or a D if the matching fields are in descending order.

Sequence-Checking of Input Files

If there is only one input file or a chaining file in the program, this specification may be used to sequence-check fields of the file to ensure that the file is in sequence. By entering the codes (M1, M2, and M3) in columns 61-62 of the Input Specification sheet, sequence-checking with respect to these fields occurs. In column 18 either A or D must be entered to specify whether the file is ascending or descending.

FILE FORMAT (COLUMN 19)

This column is used to indicate the format of the input or the output records. Enter an F in this column if the records are fixed-length. Enter a V in this column if the records are variable in length. For more information, see the Variable-Length Records section.

BLOCK LENGTH (COLUMNS 20-23)

This specification is used to indicate the block length of the input or output records. If the input records are unblocked, enter the length of the largest record in columns 20-23. If the input record is blocked, enter the size of the largest block. The block length may be from 18 to 4096. Leading zeros may be omitted.

RECORD LENGTH (COLUMNS 24-27)

This specification is used to enter the length of the logical records contained in the file. If the file contains records that are variable in length, enter the length of the largest record. (The entry must be right-justified.)

> Note 1: If the block length equals the record length, RPG will consider the file to have unblocked records for both fixed-length and variable-length records.
>
> Note 2: If the block length is greater than the record length, RPG will consider the file to have blocked records for both fixed-length and variable-length records.

MODE OF FILE PROCESSING (COLUMN 28): DIRECT ACCESS STORAGE DEVICE (DASD)

This column is used to indicate the method or mode by which the file is processed. Acceptable entries are listed here.

Entry	Explanation
L	Enter an L in this column, if a segment of the file is to be processed. The upper and lower limits of the file (to be processed) will be supplied, in this case, by a Record Address File (RAF). The RAF will be supplied by the user.
R	Enter an R in this column, if the user's records are to be processed randomly. In this case, the records to be processed will be obtained by a record-address file or a chaining file.
Blank	If no entry is made in this column for the file, the entire file will be processed sequentially.

LENGTH OF RECORD ADDRESS FIELD (COLUMNS 29-30): RECORD ADDRESS FILE (RAF) ONLY

If the file defined on this line is a record-address file, enter the number of positions that each entry in the RAF occupies.

TYPE OF RECORD ADDRESS (COLUMN 31): (DASD OR RAF FILE ONLY)

If the records from a DASD file are to be retrieved by using record keys, enter a K in this column. The K indicates that the file defined on this line will be processed by use of the record key. If the records are to be retrieved by the record identification, enter an I in this column. If the file specified on this line is a RAF file, a K or an I must be entered in column 31. Otherwise, leave the column blank.

TYPE OF FILE ORGANIZATION (COLUMN 32): (DASD ONLY)

Leave this column blank if the file is organized sequentially, or if it is an output file. Enter an I if the file has an indexed-sequential organization. Enter a D in this column if the file has direct organization.

Enter a T in this column, if the file is the output from the ADDROUT (Address Output) option of the disk sort program. (See IBM Basic Operating System/360, Sort/Merge Program (8K Disk), Form C24-3321, for an explanation of the ADDROUT option.) An example of using this option is contained in the section of this manual entitled Using the ADDROUT Option.

RULES FOR SPECIFYING MODE OF FILE PROCESSING, TYPE OF RECORD ADDRESS, AND TYPE OF FILE ORGANIZATION

1. If a direct access storage device is not used in the system, columns 28, 31, and 32 are left blank.

2. If the type of file organization is Sequential, then columns 28, 31, and 32 are again left blank.

3. If the type of file organization is Indexed-Sequential (I in column 32), column 31 must contain a K and column 28 can be either L, R, or blank.

4. If the type of file organization is Direct (D in column 32), then column 31 can contain either a K or I, and column 28 can only contain an R.

   Note: If column 32 contains D and column 31 contains K, multiple tracks will be searched by the RPG object program.

Figure 65 illustrates the code combinations possible for these three specifications.

Type of File Organization (Col. 32)	Type of Record Addresses (Col. 31)	Mode of Processing (Col. 28)
Sequential * (blank)	Not applicable (blank)	The entire file between limits will be processed. (blank)
Indexed - Sequential (I)	Record Key (K)	(The entire file will be processed). (blank)
		A segment of the file will be processed. (L) The limits to be processed are supplied by a Record Address File (RAF).
		The records will be processed randomly. (R) The addresses are supplied: (a) by a RAF file, or (b) by the data contained in the chaining field of an input record.
Direct (D) (Random)	Track Address with Record Key, or (K) Track Address with Record Identification (I)	The records will be processed randomly. (R) Conversion is required. The addresses to be converted are supplied by a RAF file or by a chaining field.

*Entries in parenthesis indicate actual specifications to be used.

Figure 65. Processing Direct Access Storage Files

OVERFLOW INDICATOR (COLUMNS 33-34)

If the file defined on the line is a printer file and overflow indicators are used, enter the overflow indicator associated with the file. A maximum of three overflow indicators is allowed. The following are permissible overflow indicators:

1. OA
2. OF
3. OV

KEY FIELD STARTING LOCATION (COLUMNS 35-38)

Enter in these columns the starting location within the record of the field specified as the Key Field. (Leading zeros may be omitted.) This entry is used only if the records are blocked records for an Indexed-Sequential file.

E (COLUMN 39)

If the file defined on the line is a chaining file, a table file, or a record-address file, enter E in column 39. The E indicates that additional information about the file will be coded on the File Extension Specification sheet. If the file is defined as an output file (O in column 15), this column is left blank.

DEVICE (COLUMNS 40-46)

Enter in these columns the input or output unit associated with the file. If the output file is a printer, enter PRINTER in columns 40-46. If the file is an input or output file and it is associated with a Card Reader or Card Punch unit, enter one of the following:

READ42 for the IBM 1442

READ01 for the IBM 2501

READ20 for the IBM 2520

READ40 for the IBM 2540

If the file is an input or output file and it is associated with a tape unit, enter TAPE in these columns. If the file is associated with a Direct Access Storage Device (DASD), enter DISK11 in columns 40-46. The entry must be left-justified.

SYMBOLIC DEVICE (COLUMNS 47-52)

If the file defined on this line is an input or an output file, enter the symbolic name of the unit. (For example: SYSIPT in 47-52).

Valid entries in these columns are the following:

Entry	Examples
SYSRDR	Card input.
SYSLST	Printer.
SYSIPT	Card or tape input.
SYSOPT	Card, disk, tape, or printer output.
SYS001 SYS254	Any input or output unit.

Note 1: Input units containing files organized in indexed-sequential or random organization must be assigned Symbolic Device codes of SYS001-SYS254.

Note 2: If the logical file extends over more than one physical unit, the device codes must be adjoining. For example, if a logical file is contained in three IBM 2311's the device codes could be SYS019, SYS020, SYS021.

LABEL (COLUMN 53)

The entries for this specification are:

Blank = No labels exist for the file.

1. The RPG processor processes no labels and makes the first data record available.
2. RPG writes the first record.

N = The file has nonstandard labels.

1. For both input and output files, RPG always provides for an exit to a user's subroutine.

2. The user must provide for the checking of header and trailer labels.

S = The file has standard labels only.

1. An exit is not given for either an input or output file.

2. User standard labels will be bypassed.

E = The file has standard labels and user standard labels.

1. For both input and output files RPG provides for an exit to a user subroutine in which user standard labels are processed.

NAME OF LABEL EXIT (COLUMNS 54-59)

Enter in these columns the name of the external routine which will process nonstandard or user-standard labels. The name must consist of alphameric characters and cannot exceed six characters in length. (This entry is required only if column 53 contains N or E.)

Note: The register conventions for exit to a user routine for label checking are the same as those specified in the section Use of Registers with one exception. Register 9 is used as the base register, and not register 15.

EXTENT EXIT FOR DAM (COLUMNS 60-65)

In IBM Basic Operating System RPG (8K), these columns are not used.

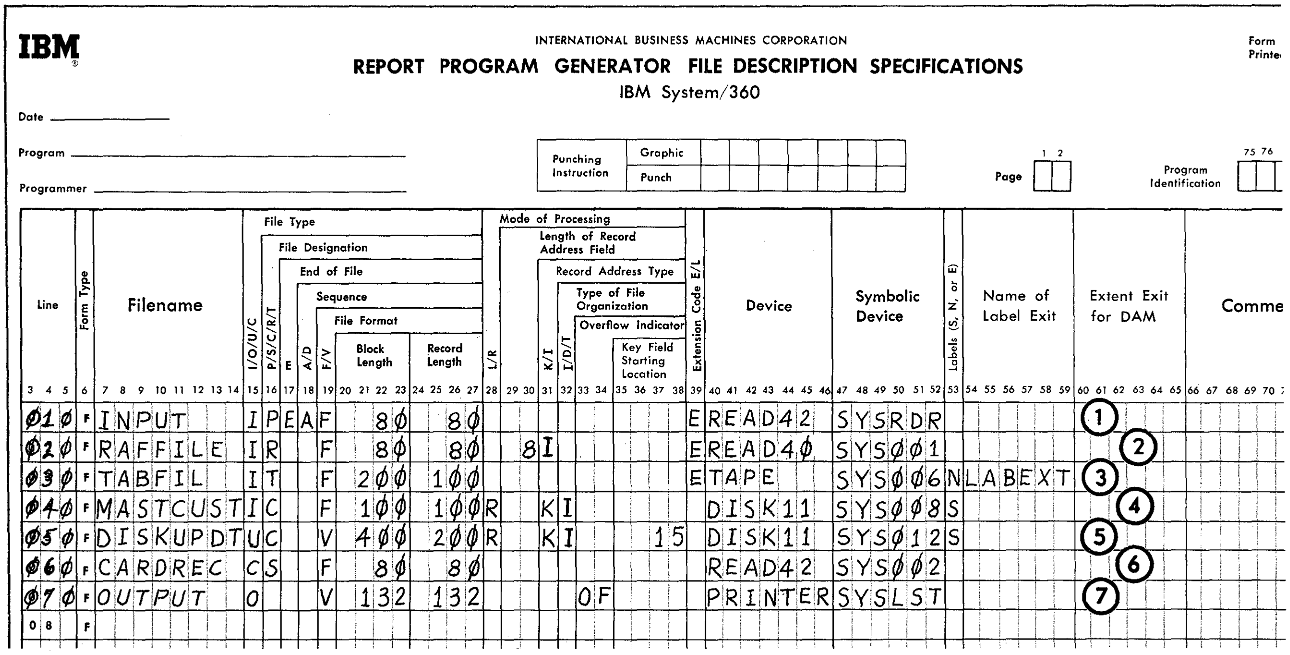
COLUMNS 66-74

Leave columns 66-74 blank, unless comments are entered in these columns.

ENTRIES ON THE FILE DESCRIPTION SPECIFICATION SHEET

Figure 66 shows several examples of entries on the File Description Specification sheet. The numbers to the right correspond to the explanation that follows.

1. The P in column 16 indicates the input file INPUT is the primary file. The E in column 17 indicates that the end-of-job condition will occur when this file is depleted. The file is ascending (A in column 18). The block length is 80,



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form
Printe

Date

Program

Programmer

Punching Instruction | Graphic | Punch

Page 1 2

Program Identification 75 76

Line (3-5)	Form Type (6)	Filename (7-14)	File Type I/O/U/C (15)	File Designation P/S/C/R/T (16)	End of File E (17)	Sequence A/D (18)	File Format F/V (19)	Block Length (20-23)	Record Length (24-27)	Mode of Processing L/R (28)	Length of Record Address Field (29-30)	Record Address Type K/I (31)	Type of File Organization I/D/T (32)	Overflow Indicator (33-34)	Key Field Starting Location (35-38)	Extension Code E/L (39)	Device (40-46)	Symbolic Device (47-52)	Labels (S, N, or E) (53)	Name of Label Exit (54-59)	Extent Exit for DAM (60-65)	Comme (66-70)
Ø1Ø	F	INPUT	I	P	E	A	F	8Ø	8Ø							E	READ42	SYSRDR			1	
Ø2Ø	F	RAFFILE	I	R			F	8Ø	8Ø		8	I				E	READ4Ø	SYSØØ1			2	
Ø3Ø	F	TABFIL	I	T			F	2ØØ	1ØØ							E	TAPE	SYSØØ6	N	LABEXT	3	
Ø4Ø	F	MASTCUST	I	C			F	1ØØ	1ØØ	R		K	I				DISK11	SYSØØ8	S		4	
Ø5Ø	F	DISKUPDT	U	C			V	4ØØ	2ØØ	R		K	I		15		DISK11	SYSØ12	S		5	
Ø6Ø	F	CARDREC	C	S			F	8Ø	8Ø								READ42	SYSØØ2			6	
Ø7Ø	F	OUTPUT	O				V	132	132					OF			PRINTER	SYSLST			7	
08	F																					

Figure 66. Entries for the File Description Sheet

and each record is 80 characters long. The E in column 39 indicates that the file will be referenced on the File Extension Specification sheet. This file is read in on an IBM 1442 Card Read-Punch. The Device code is READ42, and the Symbolic Device code is SYSRDR.

2. The record-address file defined on this line has a fixed format (column 19). It has a block length of 80. Each record is 80 characters long and the length of each record-address field is 8. The E in column 39 indicates that the File Extension Specification sheet will be used. The record address file is read into the program by an IBM 2540 Card Read-Punch. The Device code is READ40 and the Symbolic Device code is SYS001.

3. The third file defined on this sheet is a table file. The T in column 16 indicates that it is a table file. It has a fixed format, a block length of 200, and a record length of 100. Column 39 (E) indicates that it will be used on the File Extension Specification sheet. The file is read in on magnetic tape, therefore the Device code is TAPE, and the Symbolic Device code is SYS006. The N in column 53 indicates that there are nonstandard labels on the tape. This means that the nonstandard labels will be processed by the user's subroutine LABEXT (as specified in columns 54-59).

4. MASTCUST is an input file that will be processed under the control of another file. It is a chained file (C in column 16). It will be processed randomly (R in column 28). The record addresses that will be referenced by the chaining file are record keys (K in column 31), and the file is organized indexed-sequentially (I in column 32). This file is located on a direct access storage device and is given the Device code of DISK11 and the Symbolic Device code of SYS008. The S in column 53 indicates that there are standard labels on the disk.

5. The update file DISKUPDT (U in column 15) will be used for input, and it will be updated after processing of each record has been completed. It is an indexed-sequential file, and it will be processed randomly. The C in column 16 indicates that the file is a chained file. In this example, the maximum record length is 200 with a maximum block length of 400. This file is located on a direct access storage device and is given the Device code of DISK11 and the Symbolic Device Code of SYS012. The key field starting location (columns 37-38) is position 15 within the data record.

6. The file CARDREC is a combined file (column 15). Assume that the file will be used as input, and additional information will be punched in the input cards during processing. Therefore, it is a Secondary file (Column 16). This combined file is read in and punched out on an IBM 1442 Card Read-Punch. The Device code is specified as READ42 and the Symbolic Device code as SYS002.

7. The file OUTPUT is a printed report in this example. It is variable in length, and the longest record is 132 characters. The overflow indicator for this file is OF (columns 33-34). The printed report in this case has a Device code of PRINTER and a Symbolic Device code of SYSLST.

Entries made on the File Extension Specification sheet provide information to the RPG about such functions as chaining files, tables used in the object program, and information about record-address files. In the sections Using Tables in the Object Program and Processing Multiple Input Files, detailed information and examples show how to use these functions. A maximum of 25 line entries may be made on the File Extension Specification sheet.

The entries allowed on the File Extension Specification sheet are discussed in the following sections.

RECORD SEQUENCE (COLUMNS 7-8)

Chaining File

If the file defined on the line is a chaining file, enter the sequence of the file in columns 15-16 of the Input Specification sheet. The file name of the chaining file has been entered in columns 7-14 of the Input Specification sheet.

Record-Address File or Table File

Leave columns 7-8 blank if a record-address file or a table file is defined on this line.

NUMBER (COLUMNS 9-10)

Chaining Field

Enter the number of the chaining field in columns 9-10. The number of the chaining field is taken from columns 61-62 of the Input Specification sheet.

Record-Address File (RAF) or Table File

Leave columns 9-10 blank if the file defined on the line is a record-address file or a table file.

FROM FILENAME (COLUMNS 11-18)

Enter in these columns:

1. The name of the chaining file,
2. The name of the RAF, or
3. The name of the table file as defined on the File Description Specification sheet.

Chaining File. This is the file that has the data record containing the chaining field. The name of the file is taken from columns 7-14 of the File Description Specification sheet. The entry must be left-justified.

Record Address File. If the file defined on the line is a record-address file, enter the name of the RAF file in these columns. The entry must be left-justified.

Table File. Enter the name of the file that contains the table if the file defined on the line is a table file. This file has been defined on the File Description Specification sheet. The entry must be left-justified. Table files are processed in the same order in which they appear on the File Extension Specification sheet.

TO FILENAME (COLUMNS 19-26)

Chained File Name. If the file is a chained file, enter its name. The data record is obtained from this file. The entry must be left-justified.

RAF File Name. If the From Filename entry (columns 11-18) is an RAF file, enter the name of the file that contains the data record to be processed. The entry must be left-justified.

Table File Name. If the From Filename entry (columns 11-18) is a table file, enter the name of the file to be used as an output file after the table has been updated during program execution. Leave this column blank if the table file is not to be written out. A table can be put out on only one device. For additional information on tables, see Using Tables in the Object Program. The entry must be left-justified.

TABLE NAME (COLUMNS 27-32).

Columns 27-32 contain the name of either the table that contains arguments or functions. If only one table name (argument or function) is read in on one device, then it is entered in this column, and columns 46-57 are left blank. The name must be of the form TABnnn. The entries nnn may be any alphameric characters. (The entry must be left-justified.)

Direct Organization

When a file that has direct organization is processed under control of a record-address file or a chaining file, the entry for these columns is the RPG internal label of the conversion routine. The concepts of chaining and record address files have not been discussed at this point in the publication. For a complete discussion of chaining, conversion, and record-address files, see Processing Multiple Input Files.

NUMBER OF TABLE ENTRIES PER RECORD (COLUMNS 33-35)

| Enter in columns 33-35 the exact number of table entries (that is, the number of arguments or functions) that are contained in each input record. The entry must be right-justified.

NUMBER OF TABLE ENTRIES PER TABLE (COLUMNS 36-39)

In these columns, enter the exact number of table entries (the number of arguments or functions in the table) contained in the table. The entry must be right-justified.

LENGTH OF TABLE ENTRY (COLUMNS 40-42)

Enter in columns 40-42 the length of each table entry. The maximum size of an entry is 248 bytes. The entry must be right-justified.

PACKED (COLUMN 43)

If the data in the table is in the packed-decimal format, enter P in this column. Otherwise, leave this column blank.

DECIMAL POSITIONS (COLUMN 44)

If the data contained in the table is numeric, enter a zero if there are no decimal positions. Enter the number of decimal positions (1-9) in this column if the data contains decimal positions. If the field is alphameric, leave this column blank.

SEQUENCE (COLUMN 45)

If the data contained in the table is in ascending sequence, enter an A in this column. If the data contained in the table is in descending sequence, enter a D in this column. Leave this column blank if the data contained in the table is not in ascending or descending sequence or if this specification is not required.

> Note: The next five specifications (columns 46-57) are used only if alternating functions and arguments are read in on one device.

TABLE NAME (COLUMNS 46-51)

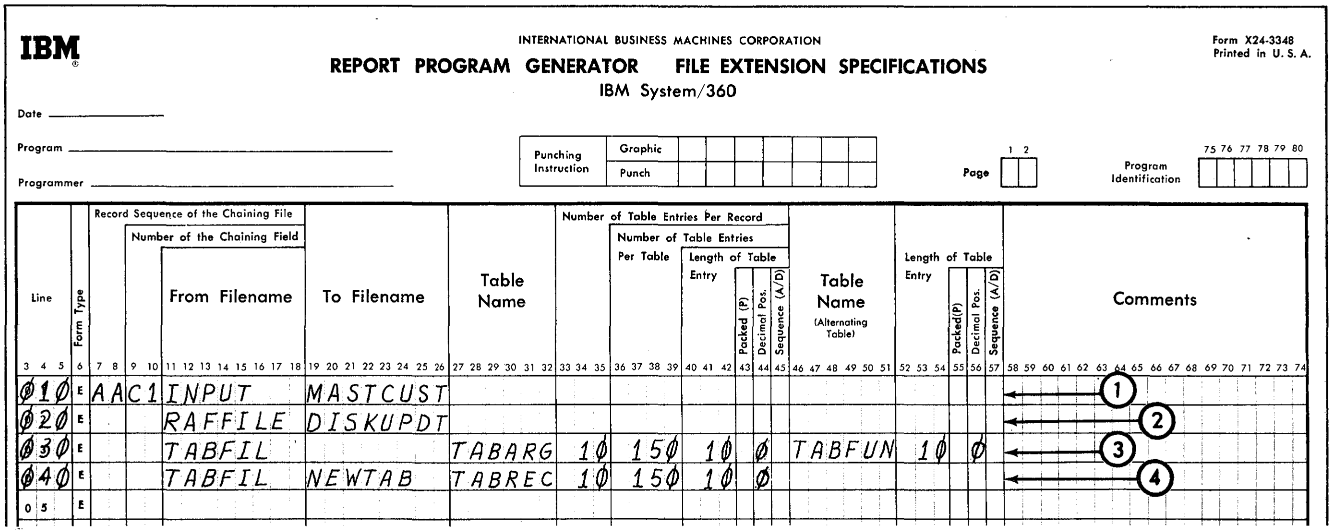
If two table names are used, enter the second table name in these columns. It must be of the form TABnnn. This specification is used only if alternating arguments and functions are used. The entry must be left-justified.

LENGTH OF TABLE ENTRY (COLUMNS 52-54)

Enter in these columns the length of each table entry. The maximum size of a table entry is 248 bytes. The entry must be right-justified.

PACKED (COLUMN 55)

Enter a P if the data in the table is in the packed-decimal format. Otherwise, leave this column blank.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Form X24-3348
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction	Graphic							
	Punch							

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Record Sequence of the Chaining File	Number of the Chaining Field	From Filename	To Filename	Table Name	Number of Table Entries Per Record	Number of Table Entries Per Table	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Table Name (Alternating Table)	Length of Table Entry	Packed(P)	Decimal Pos.	Sequence (A/D)	Comments
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
Ø1Ø	E	AA	C1	INPUT	MASTCUST													← 1
Ø2Ø	E			RAFFILE	DISKUPDT													← 2
Ø3Ø	E			TABFIL		TABARG	1Ø	15Ø	1Ø		Ø		TABFUN	1Ø		Ø		← 3
Ø4Ø	E			TABFIL	NEWTAB	TABREC	1Ø	15Ø	1Ø		Ø							← 4
0 5	E																	

● Figure 67. Entries for the File Extension Sheet

DECIMAL POSITIONS (COLUMN 56)

If the data contained in the table is numeric, enter a zero if there are no decimal positions. Enter the number of decimal positions (1-9) in this column if the data contains decimal positions. If the field is alphameric, leave this column blank.

SEQUENCE (COLUMN 57)

If the data contained in the table is in ascending sequence, enter an A in this column. If the data contained in the table is in descending sequence, enter a D in this column. Leave this column blank if the data contained in the table is not in ascending or descending sequence or if this specification is not required.

COLUMNS 58-74

Leave columns 58-74 blank, unless comments are entered in these columns.

ENTRIES ON THE FILE EXTENSION SPECIFICATION SHEET

Figure 67 shows several examples of entries that can be made on the File Extension Specification sheet. The numbers to the right of the entries correspond to these explanations:

1. In this example, INPUT is a card file containing the record key that will be used to process records in the DASD file MASTCUST. The file INPUT is the chaining file. That is, it is the file that links or chains to another file (in this case MASTCUST). The field contained in INPUT, which is used to link the two files, is the chaining field. The record sequence of the input file is taken from the Input Specification sheet. C1 is the number of the chaining field. Thus, INPUT is chained to MASTCUST by using a field defined on the input specifications that contains C1 in columns 61-62. A complete discussion of chaining may be found in the Chaining section.

2. In this example, RAFFILE is a record-address file that supplies the addresses of the records to be processed in the file DISKUPDT.

3. TABFIL is the name of a table file that contains both a table of arguments and a table of functions.

   The arguments in the file are identified by the table name of TABARG. The argument table is described in columns 33-45. There are 10 arguments in each record. The number of arguments in the table is 150, and each argument is 10 characters long. The arguments are numeric (column 44).

   The functions in the file are identified by the table name TABFUN. The function table is described in columns 52-57. Each function is 10 characters long, and each function is organized in the form: argument-function. Therefore, TABARG was specified first.

4. This example shows the specification for a table file that contains only arguments. After the table of arguments is updated the table is to be written on an output unit (card or printer.)

   TABFIL is the name of the table file.

   NEWTAB is the name given to the table file when it is being written on the output unit (output operations).

   The arguments in the file are identified by the table name of TABREC. The argument table is described in columns 33-45. Ten table entries are in each record. The number of table entries in the table is 150, and each table entry is 10 characters long. The data is numeric. TABREC is the name used during updating or during a lookup operation on the Calculation Specification sheet.

USING TABLES AND EXIT ROUTINES IN THE OBJECT PROGRAM

This section of the publication contains information on:

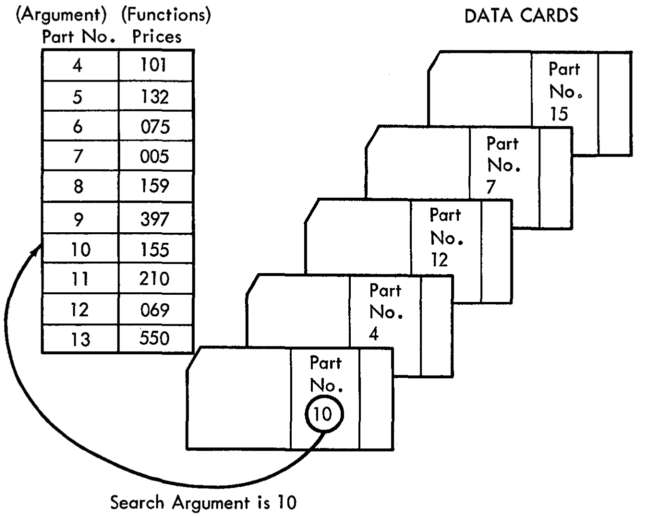
1. How to create and use tables, and

2. How to transfer control from the RPG program to a subroutine coded by the user, and how to return to the RPG program.

USING TABLES IN THE OBJECT PROGRAM

RPG enables the programmer to use tables in the object program. A table is nothing more than an arrangement of data that is searched and used by the object program. Tables are loaded into storage by the RPG object program before any files are processed. Tables can be loaded from card, tape, or disk (consecutive).

A table may consist of two parts: an argument and a function. In Figure 68 the table consists of part numbers (arguments) and prices (functions). If the price of part number 10 is wanted, the table is searched until part-number 10 is found. The corresponding function of 10 in the table is 155. The 155 represents $1.55, in this example. The number used to search the table is called the search argument. The card file in Figure 68 contains part numbers that have been ordered. The cards do not contain the prices of the parts.



(Argument) Part No.	(Functions) Prices
4	101
5	132
6	075
7	005
8	159
9	397
10	155
11	210
12	069
13	550

Figure 68. Using a Table

The part number is selected from the card by the RPG program, the table is searched, and the price is retrieved and made available for additional processing.

All entries in a table will be:

1. Arguments,
2. Functions,
3. Alternating arguments and functions, or
4. Alternating functions and arguments.

Figure 69 shows these four possibilities.

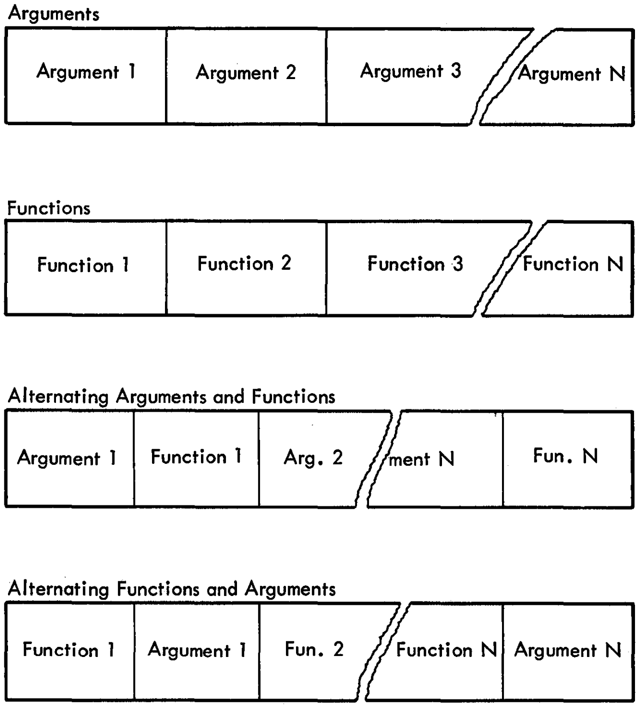


Figure 69. Four Types of Tables

RULES FOR FORMING TABLES

1. Each unit of table data is called a table entry. That is, each argument is a table entry, and each function is a table entry.

2. The collection of all arguments is assigned a name. The collection of all function entries is assigned a name. The names must be unique, and the names of the tables must contain the letters TABnnn (nnn may be any alphameric

entry). In Figure 73 the alternating table file called RATETABL is split into the collection of arguments (TABNUM) and functions (TABRAT). RATETABL is the name of the file containing the tables.

3. Although all the tables may be loaded from the same device, a unique file name must be assigned in columns 11-18 for each line entry on the File Extension Specification sheet. The tables will be loaded into storage before the object program is processed, and each line entry on the File Extension Specification sheet must have:

   a. A unique file name (columns 11-18).

   b. Entries in columns 27-45 if the table is only arguments or functions.

   c. Entries in columns 27-45 and columns 46-57 if the table consists of alternating arguments and functions.

   Note: If the program is to be compiled and executed at a later time using Operating System RPG, all tables read or written using the same I/O device must have the same file name.

Rules for Creating Records Containing Table Data

1. Each record must begin with the first table entry of that record in position 1.

2. All records must have the same number of table entries, except the last. In Figure 70, the first card in the table file has ten table entries. All subsequent card records must have ten table entries. The second card could not contain six; the third could not contain seven, for example.

3. All entries must be continuous in every record. In Figure 70, the first entry begins in position 1 and the second entry begins in position 4. No blanks can be contained between the table entries.

4. All entries belonging to a table must have the same length. In Figure 70, each argument is three positions long and each function is six positions long.

5. When alternating tables are used, each record must begin with an entry of the

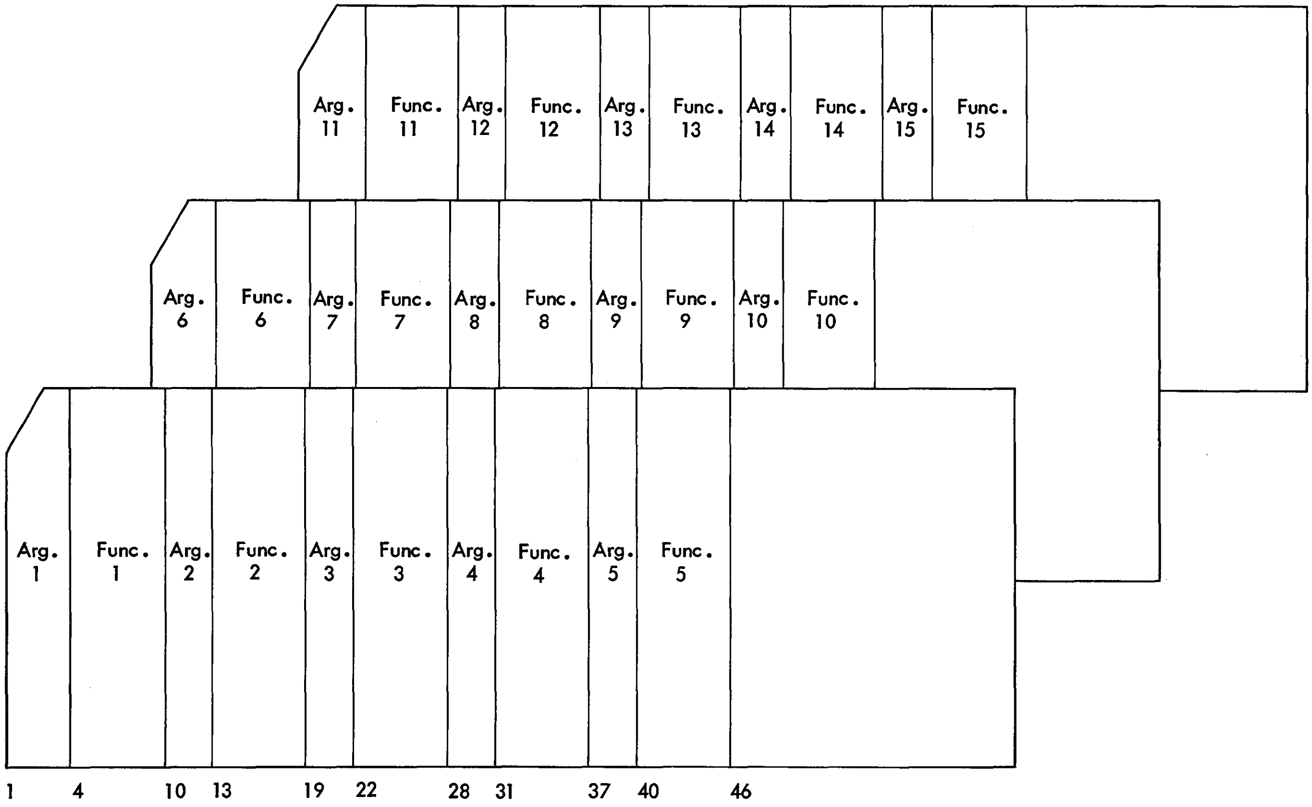


Figure 70. Table File Containing Arguments and Functions

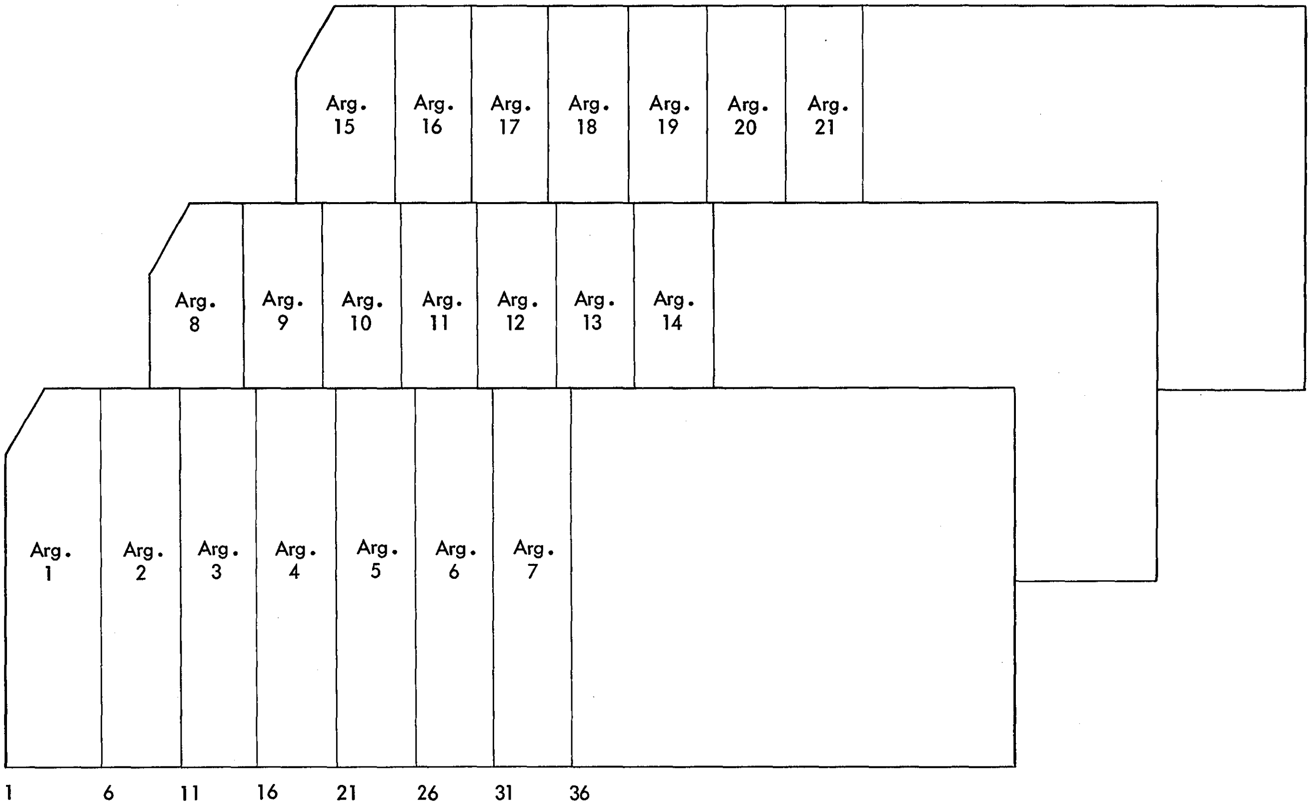


Figure 71. Table File Containing All Arguments

same type. Each record must always begin with an argument, or each record must always begin with a function as shown in Figure 70.

6. When alternating tables are used, the table entries in each record must not be split. Function 3, for example, must be in the same record as argument 3. It is not permissible for a function to appear in a different record other than its corresponding argument.

7. If a table consists of all arguments or all functions, an argument or a function must not be split. Assume that argument one, argument two, argument three, and argument four are contained in the first record. No part of argument four could overflow into the second record. Figure 71 illustrates the correct way to specify records containing arguments or functions.

8. The table may be ascending, descending, or in no sequence.

9. The records of a table must be on a sequentially organized file.

10. The table file to be loaded must contain the exact number of table entries as specified on the File Extension Specification sheet.

METHODS OF PROCESSING TABLES

The operation code LOKUP entered on the Calculation Specification sheet causes a table lookup operation to be performed.

Factor 1 contains the search argument. The search argument may be a literal or a field name.

Note: The length of the data in the argument table (table argument) must be equal to the length of the search argument. The length includes the decimal positions.

Factor 2 contains the name of the table which contains the arguments.

The result field contains the name of the table from which an associated function is to be located, if arguments and functions are used. The result field may be left blank if the user wants to determine if an argument is present in the table, but does not require the corresponding function.

Resulting indicators (columns 54-59) must always have an entry when the table lookup operation is performed. The indicators indicate the type of lookup to be performed, and the indicators are turned on whenever the condition is satisfied. The program may search for the table argument next-higher than the search argument, or it may search for the table argument next-lower than the search argument, or it may search for the table argument equal to the search argument. An entry must be made in columns 54-59. Combinations of high-equal or low-equal searches may be specified by placing indicators in the appropriate two of the three fields (columns 54-59).

The lookup operation is performed in this way:

1. The object program takes the field name or literal in Factor 1 and searches the table indicated by Factor 2. The kind of lookup is determined by the entries in the resulting indicators.

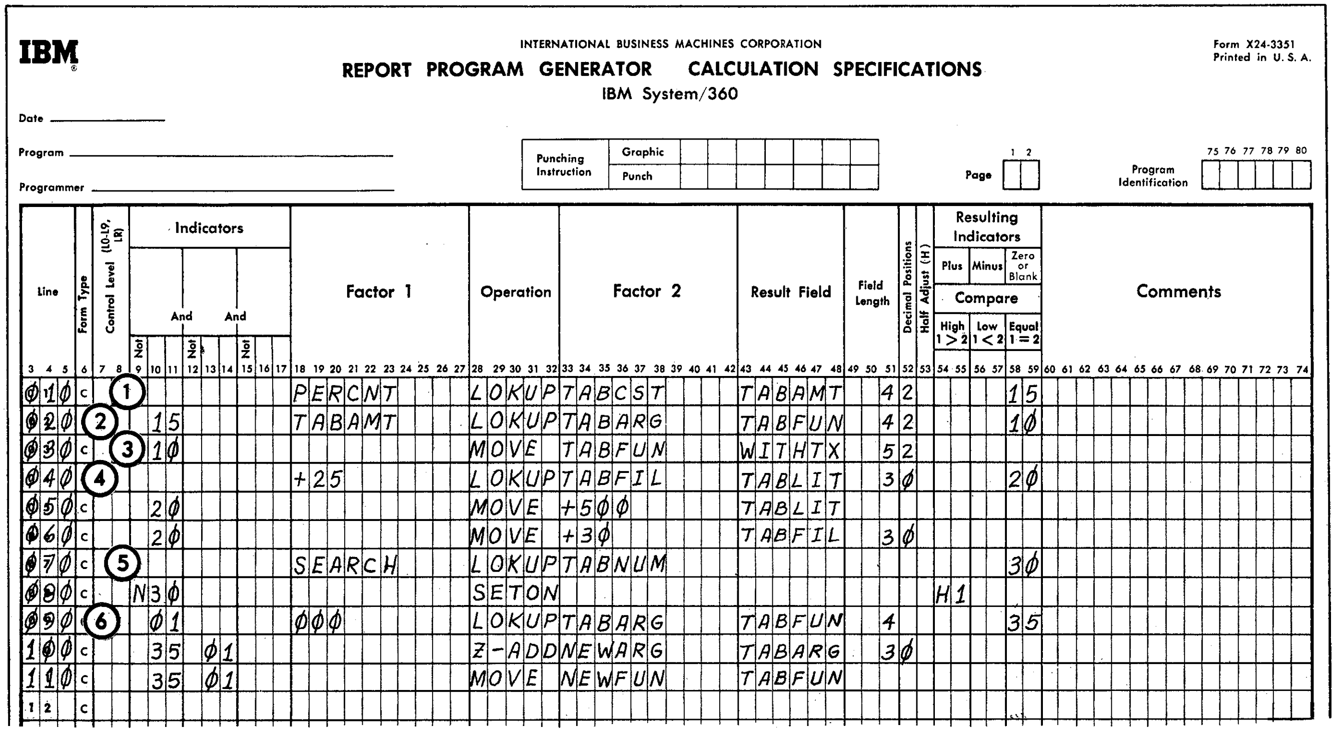
2. After the proper entry from the argument table has been found, the corresponding function from the function table (indicated by the entry in the result field) is located, and placed in the special holding area of the function table. If the proper table argument is not found, the indicators in columns 54-59 are not turned on.

Other operations may be performed using the data just found by the table lookup operation. This data is stored in the special holding area within the function table, and can be retrieved by merely using the name of the function table in either Factor 1 or Factor 2 of an operation.

Figure 72 illustrates several ways the data found by the operation may be used. The numbers on the figure correspond to this discussion.

1. Factor 1 contains the name of a field. The field PERCNT contains the search argument. The name of the table that contains the arguments is TABCST. The name of the table that contains the corresponding function is TABAMT. The program will search for the value in the argument table that is equal to the search argument (columns 58-59).

2. The data found in TABAMT from the previous operation is used as the search argument in this example. (TABAMT is the name of the table; however, Factor 1 is the function located in the special holding area of the table TABAMT). The program searches for the value in the table TABARG that is equal to the search argument (columns 58-59).



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR) (7-8)	Indicators (9-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators Plus / Compare High 1>2 (54-55)	Minus / Low 1<2 (56-57)	Zero or Blank / Equal 1=2 (58-59)	Comments (60-74)
Ø1Ø	c	1		PERCNT	LOKUP	TABCST	TABAMT	4	2				15	
Ø2Ø	2		15	TABAMT	LOKUP	TABARG	TABFUN	4	2				1Ø	
Ø3Ø	c	3	1Ø		MOVE	TABFUN	WITHTX	5	2					
Ø4Ø	4			+25	LOKUP	TABFIL	TABLIT	3	Ø				2Ø	
Ø5Ø	c		2Ø		MOVE	+5ØØ	TABLIT							
Ø6Ø	c		2Ø		MOVE	+3Ø	TABFIL	3	Ø					
Ø7Ø	c	5		SEARCH	LOKUP	TABNUM							3Ø	
Ø8Ø	c		N3Ø		SETON						H1			
Ø9Ø	6		Ø1	ØØØ	LOKUP	TABARG	TABFUN	4					35	
1ØØ	c		35 Ø1		Z-ADD	NEWARG	TABARG	3	Ø					
11Ø	c		35 Ø1		MOVE	NEWFUN	TABFUN							
1 2	c													

Figure 72. Using Table Files

3. In this example, the data obtained from the function table TABFUN from the previous lookup operation is moved to a field called WITHTX. It will be used for additional calculations.

4. In this example, the data found in the function and argument tables is updated. The literal +25 is the search argument. The table TABFIL is searched for +25 (indicated by the entry in columns 58-59). A new entry for the corresponding function of +25 is entered in TABLIT. The new function is +500; the new argument is +30.

5. In example 5, a lookup with only one argument table turns on indicator 30 if SEARCH is equal to an argument in TABNUM. If 30 is not on (N30), H1 is turned on by the SETON instruction.

6. This example illustrates the facility for adding to the table. In this example, the LOKUP operation is conditioned on indicator 01. (Indicator 01 is turned on when the input file contains records with additional table information. Each record contains the two fields, NEWARG and NEWFUN.) To determine the first vacant argument, a field of zeros is used as the search argument. (Zeros are used because the argument field is numeric. Blanks would be used if the argument field had been alphabetic.)

   If there is an equal compare, indicator 35 is turned on. Since the argument field of the table is vacant, the corresponding function field is also vacant. (It is the user's responsibility to make certain that the vacant table entries contain all zeros or blanks, depending on whether the fields are numeric or alphameric.) The new argument (NEWARG) is inserted in the TABARG field, and the corresponding new function (NEWFUN) is inserted in the TABFUN field.

RETRIEVING UPDATED TABLES

After a table has been updated, the table may be written out for later use. On the File Description Specification sheet, the programmer enters the name of the file that will contain the updated table. The file must be defined as an output file. On the File Extension Specification sheet, the programmer enters the name of the file on which the updated table will be written. He enters the name of the file in <u>To Filename</u>. The updated table file will be put onto the output file after the program has reached the end-of-job condition (LR condition).

The name of the file need not be entered on the Output-Format Specification sheet. If the updated table is to be put out on a printer, no automatic skip to a new page will be initiated by the RPG program.

The output table must have the same format as the input table. When using tape units as output devices for tables in blocked format, each output table file requires a separate tape unit.

EXAMPLE OF USING TABLES

Figures 73 and 74 illustrate an input data file, the way a table might appear, and the entries necessary on the RPG specification sheets in a program that uses tables. In this example, a card-input file contains the number of hours worked by each employee (columns 42-44) and the employee's number (columns 1-5). The RPG program takes the employee number and uses it as the search argument to find the salary rate for the employee. After the employee's rate has been determined, the rate is multiplied by the number of hours worked by the employee. The result of this operation is the amount earned for each employee.

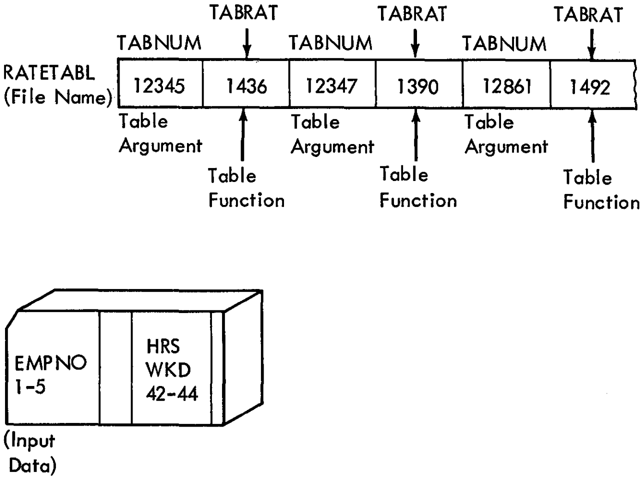


Figure 73. Using Alternating Arguments and Functions

In this example, the table consists of alternating arguments and functions. The way the table data might appear is shown in Figure 73) The name of the file that contains the arguments and functions is RATETABL. The collection of arguments is called TABNUM (the table number), and the collection of functions is called TABRAT (table rate).

Entries on the specification sheets follow.

FILE DESCRIPTION SPECIFICATION SHEET

The two files are defined on the File Description Specification sheet. The file containing the input card records is called TIMECARD. It is an input file (I in column 15); it is the primary file (P in column 16); and when the file is depleted, processing is terminated (E in column 17). The records in the file are in ascending order (A in column 18); they are fixed-in-format (F in column 19). Each record has a block length of 80 (80 in columns 22-23), and each record is 80 characters long (80 in columns 26-27).

This file is read in on the IBM 2540 Card Read-Punch, so the Device code is READ40, and the Symbolic Device code is given as SYSRDR.

The table file is defined on the line below the card-input file. The name of the file (RATETABL) is entered in Filename (columns 7-14). It is an input file (I in column 15), and the records in the file are fixed in format (F in column 19). The file has a block length of 80, and each record is 80 characters long. The E in column 39 indicates that additional information about the file is coded on the File Extension Specification sheet.

This file is read in on the IBM 1442 Card Read-punch, and therefore it is assigned the Device code of READ42. This is the second card reading device on the system so it is assigned a Symbolic Device code of SYS001.

FILE EXTENSION SPECIFICATION SHEET

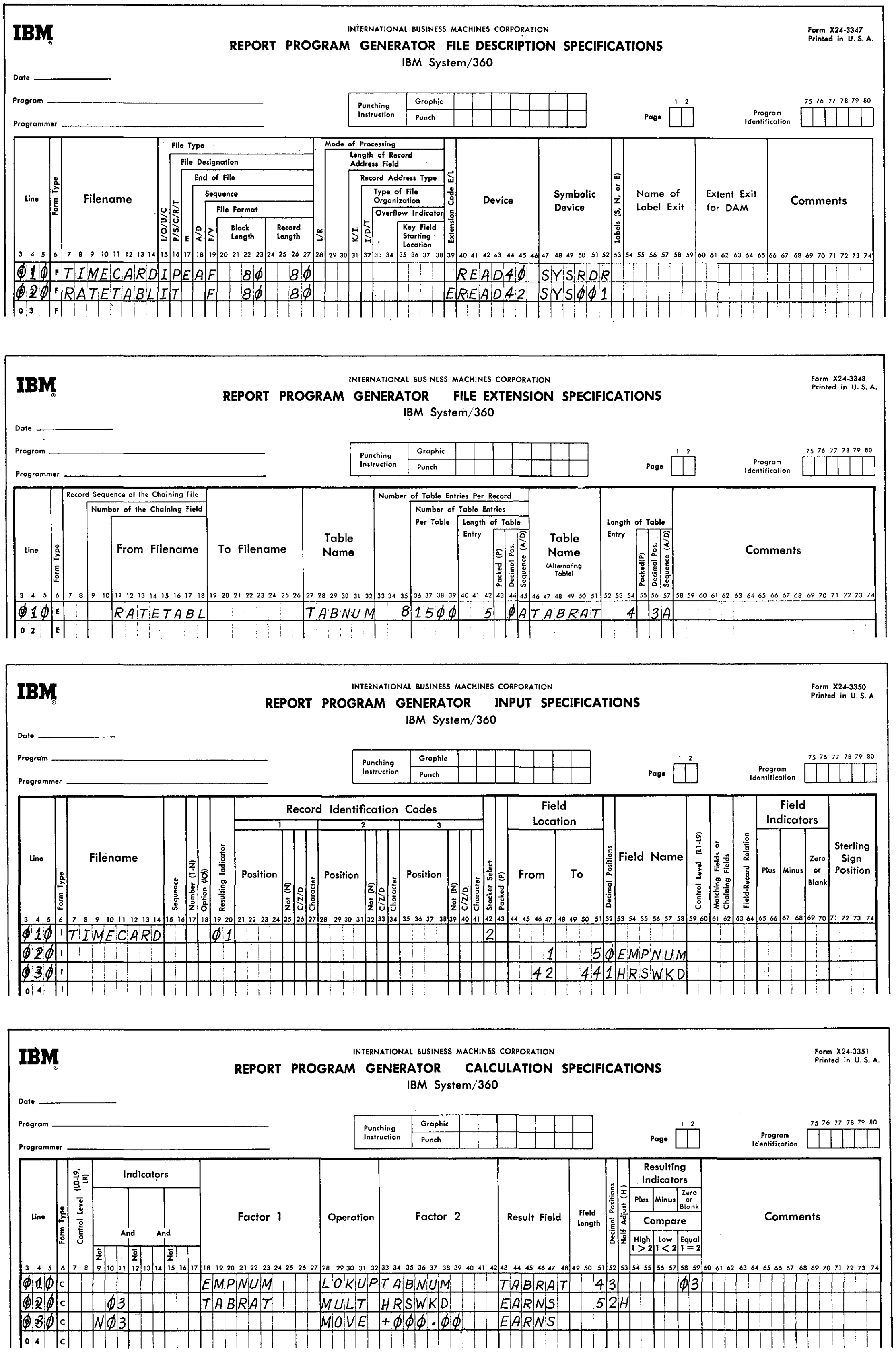
On the File Extension Specification sheet, the table file is further defined. The name of the file is entered in From Filename (columns 11-18). The collection of arguments (TABNUM) is entered in the first Table Name (columns 27-32). There are eight arguments per record (columns 34-35), and there are 1500 entries in the table (columns 36-39). Each table entry is five positions long (5 in column 42), and there are no decimal positions (0 in column 44). The table is ascending (A in column 45).

The collection of functions is described in columns 46-57. The name of the functions (TABRAT) is entered in columns 46-51. Each entry in the table is four positions long (4 in column 54), and there are three decimal positions specified (3 in column 56). The functions are in ascending order.

INPUT SPECIFICATION SHEET

The input file (TIMECARD) is described on the Input Specification sheet. The name of the file is entered in columns 7-14. The file is assigned a sequence of AA (columns 15-16), and resulting indicator 01 is turned on whenever an input record is present for processing. No record identification codes are specified because every record will be processed in the same way.

Lines 020 and 030 are used to describe the locations of the two input fields used by the program. The employee number is located in columns 1-5 of the input record, as specified by the entries in Field Location (columns 47 and 51), and the employee number is given the field name EMPNUM. The number of hours worked by the employee is found in columns 42-44 of the input record, as specified by the entries in field location. The name HRSWKD is assigned to the number of hours worked by each employee.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	Extension Code E/L	Device	Symbolic Device
Ø1Ø	F	TIMECARD	I	P	E	A	F	8Ø	8Ø		READ4Ø	SYSRDR
Ø2Ø	F	RATETABL	I	T			F	8Ø	8Ø	E	READ42	SYSØØ1
0 3	F											

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Form X24-3348
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	From Filename	To Filename	Table Name	Number of Table Entries Per Record	Number of Table Entries Per Table	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Table Name (Alternating Table)	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Comments
Ø1Ø	E	RATETABL		TABNUM	8	15ØØ	5		Ø	A	TABRAT	4		3	A	
0 2	E															

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Resulting Indicator	Stacker Select	Field Location From	Field Location To	Decimal Positions	Field Name
Ø1Ø	I	TIMECARD	Ø1	2				
Ø2Ø	I				1	5	Ø	EMPNUM
Ø3Ø	I				42	44	1	HRSWKD
0 4	I							

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators Equal 1 = 2
Ø1Ø	C		EMPNUM	LOKUP	TABNUM	TABRAT	4	3		Ø3
Ø2Ø	C	Ø3	TABRAT	MULT	HRSWKD	EARNS	5	2	H	
Ø3Ø	C	NØ3		MOVE	+ØØØ.ØØ	EARNS				
0 4	C									

● Figure 74. Coding Sheets for Table Example

CALCULATION SPECIFICATION SHEET

Three calculation specifications are shown. On line 010, EMPNUM (employee number) is used as Factor 1. The employee number is the search argument. The operation code LOKUP which is coded in columns 28-32 causes the lookup operation to be performed. Factor 2 contains the name of the collection of arguments (TABNUM) which is searched by the search argument. The result field contains the name of the collection of functions (TABRAT). Thus, this operation causes the employee number (EMPNUM) to be used as the search argument for the data contained in TABNUM. The result field is four positions long with three decimal positions. The 03 entered in columns 58-59 indicates that indicator 03 will be turned on when the search argument finds an entry in the argument table that is equal to the search argument.

The specifications on line 020 are performed when indicator 03 is on. The rate for the employee (TABRAT) which has just been located is multiplied by the number of hours worked (HRSWKD), and the result is stored in EARNS which is five positions long and has two decimal positions. The answer is half-adjusted.

If the search argument does not find an equal entry in the argument table (indicator 03 is not on), the specifications on line 030 are performed. Columns 9-11 contain the specification N03.

The literal +000.00 is then moved to the field EARNS, specifying that the employee does not have an entry in the table.

EXIT TO A USER'S ROUTINE

GENERAL INFORMATION

By use of the EXIT operation code on the Calculation Specification sheet, RPG provides the facility to transfer control from the RPG object program to some subroutine that has been coded independently. A subroutine might be a standard routine (such as a state withholding tax) or it might be a routine that performs a function not easily accomplished using RPG (such as a square-root computation). The subroutine, written in the Assembler language, is coded by the user, and entries made on the calculation sheet enable the programmer to:

1. Exit from the RPG program to the subroutine,
2. Execute the subroutine, and
3. Return to the main program after the subroutine has performed its function. Each subroutine must be assembled as a separate module.

HOW TO CODE EXIT

On the Calculation Specification sheet, the EXIT operation can be a conditional operation. 7-8, 9-11, 12-14, or 15-17, the EXIT occurs when the designated conditions are satisfied. If no indicators are used, the EXIT occurs every time the detail calculations are performed. Columns 28-31 must contain the operation code EXIT, and Factor 2 must contain the label of the user's subroutine.

POSITION OF EXIT IN THE CALCULATION SPECIFICATION ENTRIES

The following results are obtained depending on the location of the EXIT code on the Calculation Specification sheet.

Calculation Entry	When the EXIT Will Occur
1. First Detail	At the end of the data routine (after the data is extracted from the input record).
2. Last Detail	Immediately before heading records are written.
3. First Total	At the end of the input routine (after the record-type has been determined and the control-field break has been tested).
4. Last Total	Immediately before the total records are written.

GENERAL RULES FOR USING EXIT

RPG provides the facility for the subroutine to test indicators and use fields that have been defined in the RPG program. RPG also provides the facility for the RPG program to use fields that have been defined in the subroutine. These two facilities are provided by using the two operation codes RLABL and ULABL.

If the user has defined a field in the subroutine, and this field is to be used in this RPG program, he must code:

1. ULABL in the operation-code columns.

2. The label of the field in Result Field.

3. The length of the field in Field Length.

4. The decimal length in Decimal Position, if applicable.

If the user has defined a field in the RPG program and this field is to be used in the subroutine to which the EXIT will occur, he must code:

1. RLABL in the operation code.

2. The label of the field in Result Field.

When executing the subroutine, the user may need to reference the resulting indicators that are used in the RPG portion of this program. To do this, the user must code:

1. RLABL in the operation code columns,

2. INnn in the result field. The nn represents the specific indicator that the user wants to test in the subroutine. Therefore, if MR was to be tested in the subroutine, he would code INMR in the result field.

Indicators defined in the subroutine cannot be used or referenced in the basic operating system RPG program.

USE OF REGISTERS

The way in which registers are used by the programmer is strictly defined. These rules must be followed:

1. The register that contains the base address of the subroutine is register 15. The first instruction of the programmer's subroutine must load the base address of the subroutine in register 15.

   Note: The base register for an exit to a user's subroutine for label checking is register 9, and not register 15.

2. When control of the program passes from the RPG program to the subroutine, the address of the RPG instruction to which the subroutine must return is stored in register 14.

3. The RPG instruction to which the subroutine returns is the instruction that follows the EXIT operation.

4. If registers are used within the subroutine, the contents of the registers the programmer intends to use must be stored before the subroutine is executed.

5. Before the subroutine transfers back to the RPG program, the registers must be restored to their original contents.

6. The general registers available for the user's purposes are registers 2, 8, 9, 10, and 11. Registers 3-7 can be used only if no RLABL's are specified in the user's subroutine. The remaining registers cannot be used.

USING INDICATORS AND FIELDS IN THE EXIT ROUTINE

If in the exit subroutine, the user sets on, sets off, or tests resulting indicators, he must observe the following rules:

1. To set on a resulting indicator, set the data located at INnn to the hexadecimal F0.

2. To set off a resulting indicator, set the data located at INnn to the hexadecimal 00.

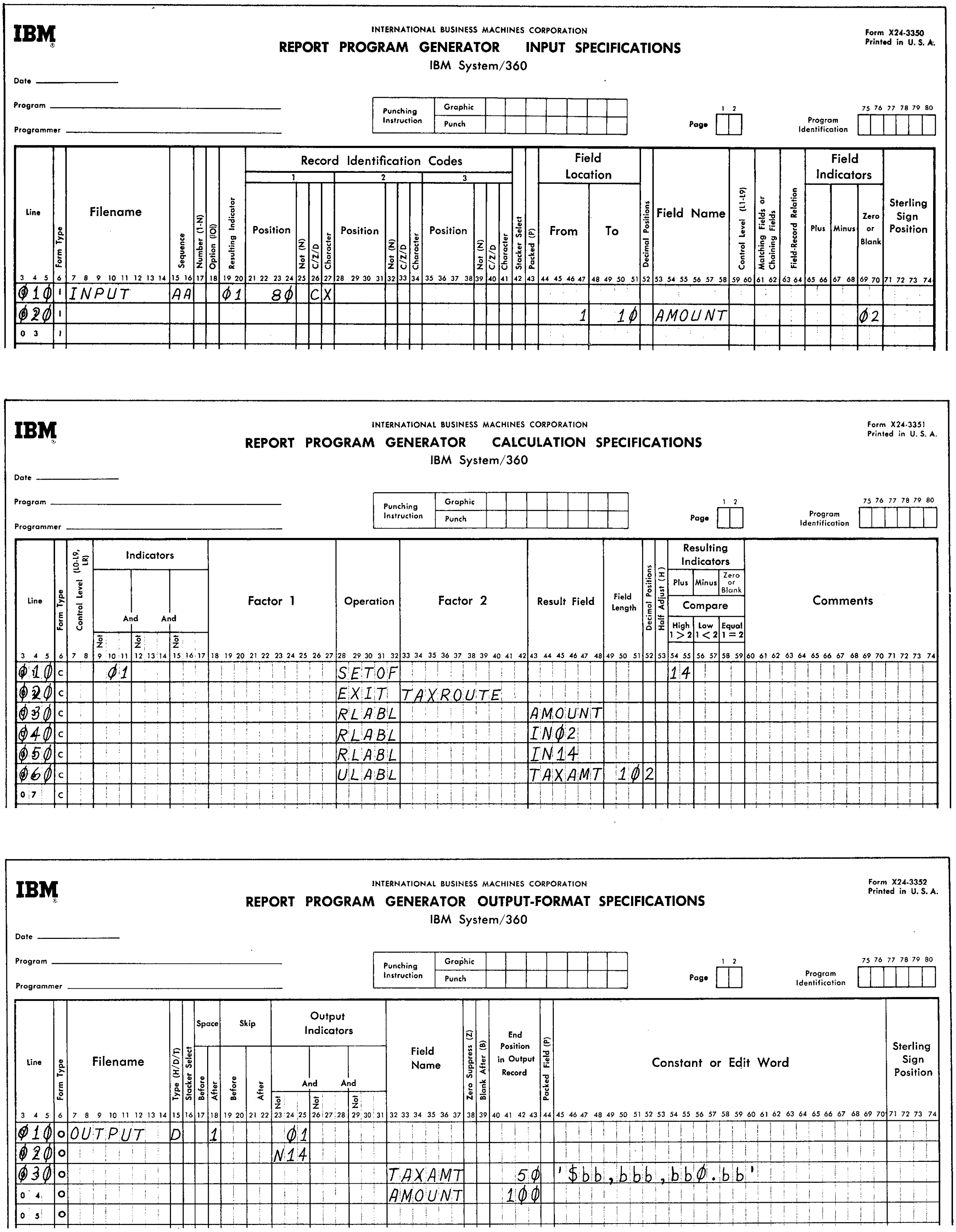
3. To test resulting indicators:

   a. If on, the data at INnn, will be the hexadecimal F0.

   b. If off, the data at INnn will be the hexadecimal 00.

If numeric data from the RPG object program is used in the subroutine, it will be in the packed-decimal format. If the numeric data from the subroutine is supplied to the RPG object program, it must be in the packed-decimal format.

Figure 75 shows the coding steps necessary to implement the EXIT routine.



IBM INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Resulting Indicator (19-20)	Record Identification Codes 1 Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	2 Position (28-31)	Not (N) (32)	C/Z/D (33)	Character (34)	3 Position (35-38)	Not (N) (39)	C/Z/D (40)	Character (41)	Stacker Select (42)	Packed (P) (43)	Field Location From (44-47)	To (48-51)	Decimal Positions (52)	Field Name (53-58)	Control Level (L1-L9) (59-60)	Matching Fields or Chaining Fields (61-62)	Field-Record Relation (63-64)	Field Indicators Plus (65-66)	Minus (67-68)	Zero or Blank (69-70)	Sterling Sign Position (71-74)
Ø1Ø	I	INPUT	AA			Ø1	8Ø		C	X																					
Ø2Ø	I																				1	1Ø		AMOUNT						Ø2	
0 3	I																														

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR) (7-8)	Indicators Not (9-11)	And Not (12-14)	And Not (15-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators Plus / Compare High 1 > 2 (54-55)	Minus / Low 1 < 2 (56-57)	Zero or Blank / Equal 1 = 2 (58-59)	Comments (60-74)
Ø1Ø	C		Ø1				SETOF						14			
Ø2Ø	C						EXIT	TAXROUTE								
Ø3Ø	C						RLABL		AMOUNT							
Ø4Ø	C						RLABL		INØ2							
Ø5Ø	C						RLABL		IN14							
Ø6Ø	C						ULABL		TAXAMT	1Ø	2					
0 7	C															

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date
Program
Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators Not (23-25)	And Not (26-28)	And Not (29-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)	End Position in Output Record (40-43)	Packed Field (P) (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
Ø1Ø	O	OUTPUT	D			1			Ø1									
Ø2Ø	O								N14									
Ø3Ø	O											TAXAMT			5Ø		'$bb,bbb,bbØ.bb'	
0 4	O											AMOUNT			1ØØ			
0 5	O																	

Figure 75. Exit to a Subroutine

1. An input file of the name INPUT turns on resulting indicator 01 if an X is in position 80.

2. If the field AMOUNT is blank, field indicator 02 is turned on.

3. Whenever indicator 01 is on, the calculation specifications entry EXIT causes the program to exit to the user's subroutine called TAXROUTE.

   The operation preceding EXIT defines (and sets off) an indicator that can be used in the subroutine.

4. Within the subroutine, the user wants three things: the AMOUNT field, and the indicators 02 and 14. RLABL enables the subroutine to reference the AMOUNT field, and IN02 enables the subroutine to test indicator 02. IN14 enables the subroutine to utilize indicator 14 in the subroutine.

5. In the subroutine, assume there is a field (TAXAMT) that the user wants to use on the output-format specifications. If the field TAXAMT in the subroutine is blank, the subroutine turns on indicator 14. TAXAMT is referenced in the output specifications, and it will not be printed if indicator 14 is on. If the user chooses, he can use it later in the calculation specifications. The entry ULABL enables the field TAXAMT to be referenced by the RPG program.

6. On the output sheet, TAXAMT is treated as a field.

This section of the manual is for readers not familiar with disk storage operations. It contains a general description of data organization and retrieval, and defines some of the terms you may encounter in related literature.

INTRODUCTION AND TERMINOLOGY

The distinction between two concepts is important: file organization and file processing.

File Organization is the method of arranging data records on a direct access storage device. A file is organized during the development stages of the application.

File Processing is the method of retrieving data records from the file.

To achieve the most efficient use of the System/360 components, carefully consider the relationship between how a file is organized and how to retrieve records from it. This is particularly important when designing data files for storage in a direct access storage device, such as the IBM 2311.

The method of organization best suited to a particular file of disk records depends upon many factors. These factors must be analyzed for each file in each particular application. Frequently, you can use more than one method of processing on the same file. For example, records within a file might be processed at random during an updating run, and sequentially during a billing run.

LOGICAL FILE VS PHYSICAL UNIT

It is important to distinguish between a logical file and the physical unit used to store the file. A logical file is a group of related data records, such as a payroll file.

A physical unit for storage of data records could be an IBM 2400-series Magnetic Tape Unit, an IBM 2311 Disk Storage Drive, or an IBM 2540 Card Read-Punch.

A logical file may occupy part of a disk storage drive, an entire disk storage drive, or more than one disk storage drive. The location of the logical file in disk storage is defined by its lowest and highest addresses. This area is the extent area. One logical file can occupy more than one extent area. The extent areas do not have to be adjoining.

DATA FILES

Data files are recorded on such media as: paper, cards, tape, or disk packs. Data files consist of a number of individual records that range from a few records up to thousands or millions of records.

RECORD

A record can be defined as a collection of information consisting of alphameric and/or nonalphameric characters related to a common identifier. The common identifier is known as a record's control field, or key. Usually, one of the fields within a record identifies the record. For example, man number could be the key or identifier for a payroll record.

The size or length of records varies from file to file, and can be from eighteen characters to thousands of characters.

A single record usually includes one or more logical data fields. A data field is a sequence of one or more characters which is treated as a processing unit of information. An individual data field is normally identified by its location within a record.

The logical structure of records and of fields within records is important in high-speed recording media such as magnetic tape and disks. This logical structure is strongly affected by whether a record is of fixed or variable length.

FIXED-LENGTH RECORDS

In fixed-length record files, all records are allocated the same number of character storage positions. Identical data fields are present in every record, whether they are used or not. The control field (key) is usually the first field present in a record.

In many applications, the fixed length records would make inefficient use of file storage space. For example, a fixed record length of 850 positions would waste many storage positions, if the average record length is 230 positions and the minimum length is only 100 positions.

Situations such as this require the development of space-saving techniques based on varying the number of storage positions allocated to data records.

VARIABLE-LENGTH RECORDS

Completely variable-length records are sometimes developed for more efficient use of storage. In this approach, the data portion of the record may be of any length, but the key (control field) size is constant. A record-length character-count field in each record shows the length of a variable-length record.

BLOCKING RECORDS

The length of individual data records varies with type of data and the application that requires such data. The format of a data record is significant to the efficient use of the various storage media available on the System/360. One important element in the design of data records involves blocking and deblocking. Input/output units (storage media) are relatively inefficient when used to record short blocks of information. To increase the efficiency of input/output units, data records are assembled into blocks of records whose sizes are convenient and efficient for processing.

Each physical record on either tape or disk requires interrecord gaps. These gaps are blank areas used to distinguish beginning and ending points of a record. If records are blocked before loading onto a tape or disk, many of these gaps can be eliminated. Variable length blocks are permitted in basic operating system RPG. The length of a variable-length block is indicated by a block-length character-count field present in each block. (See Variable-Length Records).

The Basic Operating System handles the blocking and deblocking of records so the user need only design the most efficient blocking factor for his particular data file and equipment specifications. In the Basic Operating System, only the input records for Indexed-Sequentially or Sequentially organized files can be blocked.

FILE ORGANIZATION

Data records should be organized and stored to facilitate subsequent processing. The two types of file organization are: sequential and random.

SEQUENTIAL ORGANIZATION

The logical sequence of records in this file depends upon a significant key (control field) appearing in the records. To establish a sequentially organized file, sort and then store the records in key sequence. This provides for records with successively higher (or lower) keys (control numbers) to have successively higher physical address numbers. Cards and tape files are always organized in this serial manner and usually are considered as one continuous "string" of records. Another variation of sequential organization is known as Indexed-Sequential organization.

Indexed-Sequential Organization

In this type of file organization also, the sequence of records depends upon a key (control) field. The records are stored sequentially in the file. This variation of file organization differs form Sequential organization in two ways:

1. The records may be retrieved from the file sequentially or in a random sequence.

2. Only records with transaction activity need be retrieved.

These differences occur because Indexed-Sequential organization uses index tables which indicate to the program the general location of the records. Thus, the program does not have to step through the file, record after record, to locate a specific record.

The index tables (prepared and maintained by the Basic Operating System) are analogous to the index card file in a library. If you know the name of a book, or the author, you can look in the index and find the location of the book in the book files.

For example, this might be an address (Catalog Number) of 426.25. You would then go to the book shelves, and (if it was your first time in the library) start at the first row of the book files and proceed through the rows until you found the shelf that contained 426.25. Usually, each row contains a sign to indicate the beginning and ending numbers of all books in that particular row. This, as you proceed through the rows, you would compare 426.25 with the numbers posted on each row. Assume that one row was labeled 300.88-550.00. You would then search that row for the shelf that contained the book. The shelves (like the rows) might also contain number ranges to indicate their contents. In this case, you would scan the shelf numbers until you found something like "424.00-440.96". Then you would look at individual book numbers on that shelf until you found "426.25".

The RPG program uses index tables in much the same way to locate records organized in an Indexed-Sequential file.

RANDOM ORGANIZATION

In random file organization, the records are generally not stored in the sequence of their keys (control numbers). A randomizing formula converts the record key to a numerical address (physical address) of the storage device. The record is stored at the physical address developed by the randomizing formula. In effect, a file of records will be scattered throughout an entire disk file.

FILE PROCESSING

For the three methods of file organization (sequential, indexed-sequential, and random) there are three methods of file processing:

1. Consecutive processing of sequentially organized files.

2. Sequential processing of indexed-sequentially organized files.

3. Random processing of indexed-sequentially organized files and randomly organized files.

CONSECUTIVE PROCESSING

In consecutive processing of a sequentially organized file, every record in a file is examined, and each successive record in the physical file is processed in order. For example, in a card file, the card records are processed in the order that the cards are fed into the system. The 14th card in the file could not be processed until after the 13th card had been processed.

SEQUENTIAL PROCESSING

Sequential processing of an indexed-sequentially organized file has two variations.

1. An entire logical file is processed. For example, the physical unit consists of payroll records in cylinders 0 to 42 and inventory records in cylinders 43 to 99. Only the logical (payroll) file might be processed.

2. Only a segment of a file is processed. For example, a payroll file is to be updated with new pay increases. The payroll file is in sequence by department, and each week the pay raises for various departments become effective. Therefore, on each week's processing, only segments of the payroll file are updated. The updating is accomplished by reading in a card file that contains the limits of the file to be processed. One such card record might indicate that the records for departments 26-41, are to be updated, another the records for departments 76-80, etc.

RANDOM PROCESSING

In random processing, the sequence of processing has no relationship to the sequence the data is stored in the file. The data file could be organized in either a random or an indexed sequential order. This processing is sometimes called direct.

Indexed-Sequential Files

To find a random record in an Indexed-Sequential file, an index or series of indexes is first scanned to localize the area of search by determining the track that contains the record. The index is a sequential list of the key records (of the data) with corresponding track addresses. The entire track is then scanned to find the individual record to be processed. This kind of processing is referred to as processing in a random sequence with record keys (there is no conversion).

This type of processing is analogous to directing someone to a house location. "the Martin family lives on Harrison Street", (a track address), "and their house number is 4216" (a key).

Random Files

To find a random record in a random file, compute the track address by the same randomizing formula used to load the file of records. You can make direct access to the record. Index tables are not required. This kind of processing is called processing in a random sequence using keys or record identification (ID). The record identification indicates only the location of the record on the track. For example, the 2nd, 12th, 18th, etc. record on the track. The program makes no comparison of key (control field) data when a record number is provided.

This type of processing can also be compared to directing someone to a house location. "The Martin family lives on Harrison Street", (a track address), and their house is the 5th house from the beginning of the street". (The "5th" is the record identification.)

If random processing is performed with key field only, the user supplies track address and record key field. The program then searches the specified track for the record with the corresponding record key.

If an address is assigned to more than one record, these duplicate addresses (synonyms) will not be handled by RPG.

FILE PROCESSING IN RPG

RPG object programs process the following input file organizations:

1. Sequential.
2. Indexed-Sequential.
3. Direct.

This applies to update files as well. RPG will create sequential output files only.

SEQUENTIAL ORGANIZATION

RPG object programs process sequential files by processing the entire file from the beginning to the end. The records may be fixed or variable length and blocked or unblocked.

INDEXED SEQUENTIAL ORGANIZATION (DIRECT ACCESS STORAGE DEVICE)

RPG processes Indexed-Sequential files in three ways:

1. Sequentially, by processing the entire file.
2. Sequentially, by processing between the limits of the file.
3. Randomly, by processing records on the file in a random order.

The records may be fixed-length and blocked or unblocked.

DIRECT FILES (DIRECT ACCESS STORAGE DEVICE)

A direct file will be processed randomly. The records must be fixed-length and unblocked. The two major sections (Processing Single Input Files, and Multiple Input Files) describe disk storage processing in the RPG program. In these sections, processing methods are described in detail including specific examples of methods and of coding specifications.

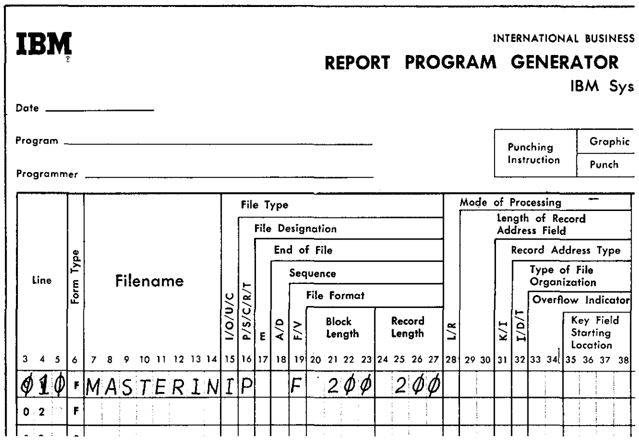
In this section, the methods of retrieving records from one input file are discussed. Three types of file organization can be processed: Sequential, Indexed-Sequential, and Direct.

If there is only one input file, it must be designated as the primary file by entering a P in column 16 of the File Description Specification sheet.

SEQUENTIAL ORGANIZATION

Sequential files may be considered as "tape-like". The records on the file will be made available for processing in the same order in which they are located on the medium. The file might be contained in cards, on magnetic tape, or on a DASD. The entire file is processed, beginning with the first record and continuing until the file is depleted. At this point, an end-of-file condition occurs, which can be interrogated by RPG by specifying an E in column 17 of the File Description Specification sheet. In the case of DASD, the extent of the file is obtained from the operating system.

Figure 76 shows the coding on the File Description Specification sheet necessary to process a sequential file. The name of the file is MASTERIN. The records are fixed in length, and the block length is 200. Each record is 200 characters long. A blank in column 32 indicates that the file is a sequential file.



IBM

INTERNATIONAL BUSINESS
REPORT PROGRAM GENERATOR
IBM Sys

Date ____________

Program ____________

Programmer ____________

Punching Instruction	Graphic
	Punch

Line (3 4 5)	Form Type (6)	Filename (7 8 9 10 11 12 13 14)	File Type: I/O/U/C (15)	File Designation: P/S/C/R/T (16)	End of File: E (17)	Sequence: A/D (18)	File Format: F/V (19)	Block Length (20 21 22 23)	Record Length (24 25 26 27)	Mode of Processing: L/R (28)	Length of Record Address Field (29 30)	Record Address Type: K/I (31)	Type of File Organization: I/D/T (32)	Overflow Indicator (33 34)	Key Field Starting Location (35 36 37 38)
ϕ1ϕ	F	MASTERIN	I	P			F	2ϕϕ	2ϕϕ						
0 2	F														

Figure 76. Coding a Sequential File

INDEXED-SEQUENTIAL ORGANIZATION

An indexed-sequential file can only be on a direct-access-storage device (such as disk or drum). For this organization method, an index table associated with the file contains a record key for every track of the file and the number of the track. For ascending files, the index table contains the highest record key on the track. For descending files, the index table contains the lowest record key on the track. An I must be entered in column 32.

For this organization, there are three methods of processing:

1. Processing the entire file sequentially.
2. Processing part of the file sequentially.
3. Processing the file randomly.

PROCESSING THE ENTIRE FILE SEQUENTIALLY

The object program obtains the limits of the file from the operating system, and the entire file is processed sequentially by record key in ascending or descending record-key sequence. Column 28 of the File Description Specification sheet must be blank.

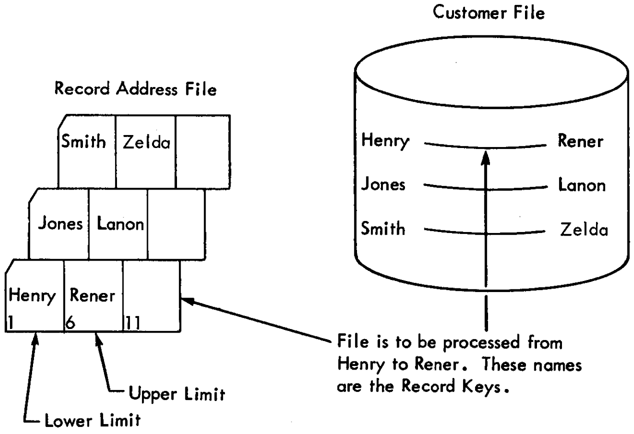
PROCESSING PART OF THE FILE SEQUENTIALLY

If only a part of the entire file is to be processed, the object program must be supplied with both the low and high keys that describe which part of the file will be processed. Column 28 must contain L.

An auxiliary file is used to supply these limits. This file is called a Record Address File (RAF). The RAF does not contain data to be processed. It contains the record keys (in this case the limits) of the data records which will be processed.

The object program obtains the limits to be processed from a record contained in a RAF and then processes all the data between those limits. The object program then reads another record in the RAF, and the procedure is continued until the RAF is depleted.

In Figure 77, the first card of the record-address file shows that the object program is to process from HENRY to RENER. The second card shows that the object program is to process from JONES to LANON. The third card shows that the third part of the file to be processed ranges from SMITH to ZELDA. Thus, the data records that the program processes are contained within these limits. In Figure 77, the record keys are customer names.



File is to be processed from Henry to Rener. These names are the Record Keys.

Figure 77. Contents of the Record Address File

The record-address file in this example is nothing more than a file which supplies limits of the file to be processed. Rules for forming record-address files are contained in Creating Record Address Files in this publication.

Figure 78 illustrates the coding required on the File Description and File Extension Specification sheets when limits of an indexed-sequential file are to be processed. In Figure 78, on the File Description Specification sheet DISKIN is the name of the input file. The records are fixed-length, and the block length is 150. Each record is 150 characters long. Limits of the file are to be processed as indicated by the L in column 28. The K in column 31 indicates that the record key will be used to obtain the records from the file.

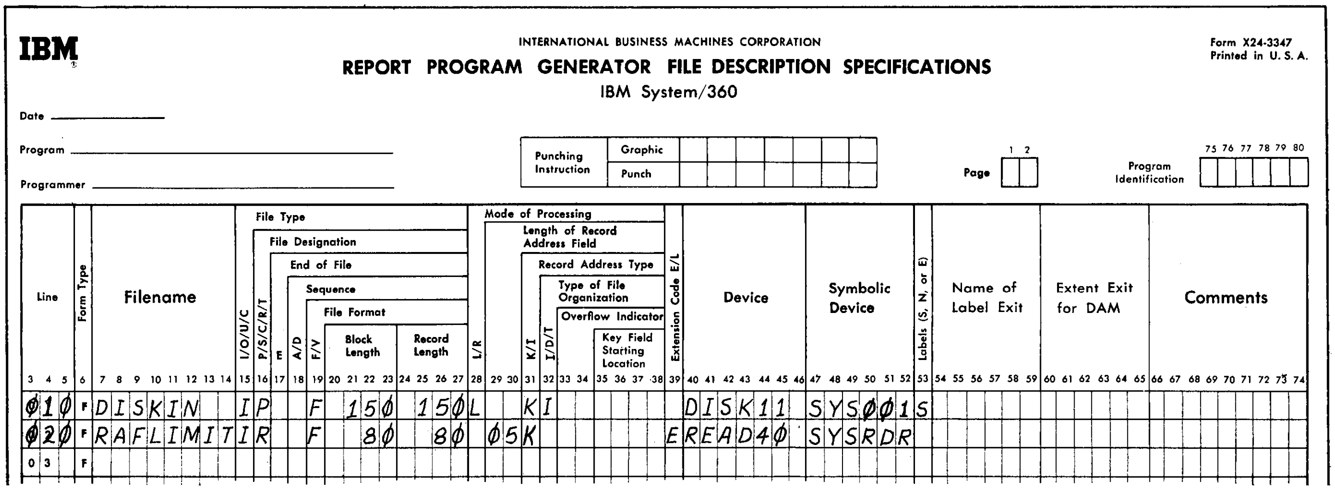
The record-address file (RAFLIMIT) is also an input file. The R in column 16 indicates that it is a RAF file. It is fixed-length; it has a block length of 80, and each record is 80 characters in length. The length of each record address field is 5 characters.

Because the File Extension Specification sheet is required to relate the RAF file to the DASD file, an E is entered in column 39. On the file extension sheet the coding illustrates that RAFLIMIT is to provide the addresses for DISKIN.

PROCESSING THE FILE RANDOMLY

A K in column 31 indicates that record keys are used to obtain the records. An indexed-sequential file may be processed randomly by supplying a RAF. Instead of supplying the limits (as in the case of sequential processing), the RAF contains the record key of each record of the file to be processed.

Figures 79 and 80 illustrate the specifications for this type of organization. In this example, the first record from the record-address file is read. The first RAF entry ADAMS is used to obtain the data record whose record key is ADAMS. The record is retrieved and is processed. The next entry, CABOT, is then used to find the next data record.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date ______

Program ______

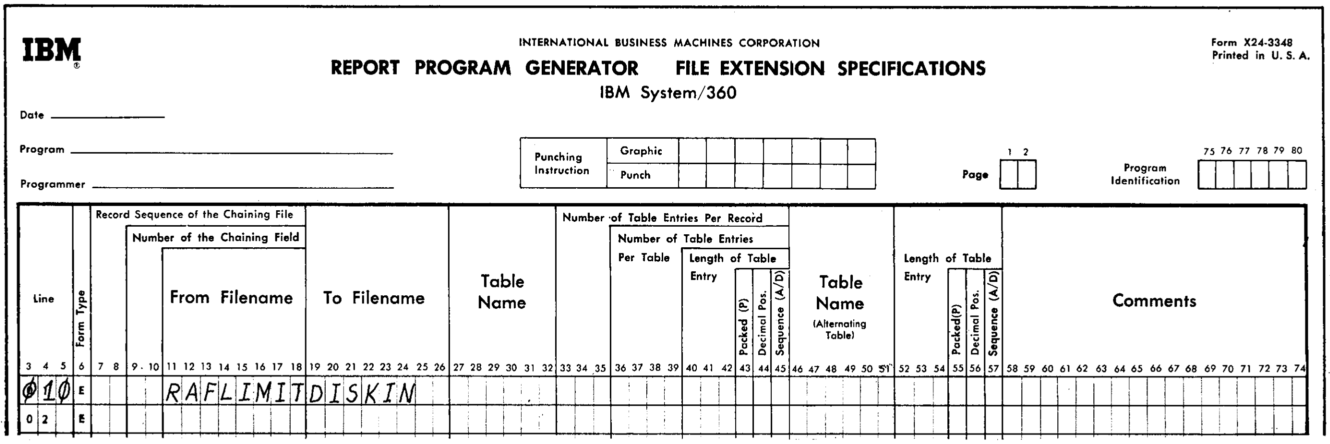
Programmer ______

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	I/O/U/C (15)	P/S/C/R/T (16)	E (17)	A/D (18)	F/V (19)	Block Length (20-23)	Record Length (24-27)	L/R (28)	Length of Record Address Field (29-30)	K/I (31)	I/D/T (32)	Overflow Indicator (33-34)	Key Field Starting Location (35-38)	Extension Code E/L (39)	Device (40-46)	Symbolic Device (47-52)	Labels (S, N, or E) (53)	Name of Label Exit (54-59)	Extent Exit for DAM (60-65)	Comments (66-74)
ϕ1ϕ	F	DISKIN	I	P			F	15ϕ	15ϕ	L		K	I				DISK11	SYSϕϕ1	S			
ϕ2ϕ	F	RAFLIMIT	I	R			F	8ϕ	8ϕ		ϕ5	K				E	READ4ϕ	SYSRDR				
0 3	F																					



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Form X24-3348
Printed in U. S. A.

Date ______

Program ______

Programmer ______

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Record Sequence of the Chaining File (7-8)	Number of the Chaining Field (9-10)	From Filename (11-18)	To Filename (19-26)	Table Name (27-32)	Number of Table Entries Per Record (33-35)	Number of Table Entries Per Table (36-39)	Length of Table Entry (40-42)	Packed (P) (43)	Decimal Pos. (44)	Sequence (A/D) (45)	Table Name (Alternating Table) (46-51)	Length of Table Entry (52-54)	Packed(P) (55)	Decimal Pos. (56)	Sequence (A/D) (57)	Comments (58-74)
ϕ1ϕ	E			RAFLIMIT	DISKIN													
0 2	E																	

● Figure 78. Processing a Part of the Indexed-Sequential File

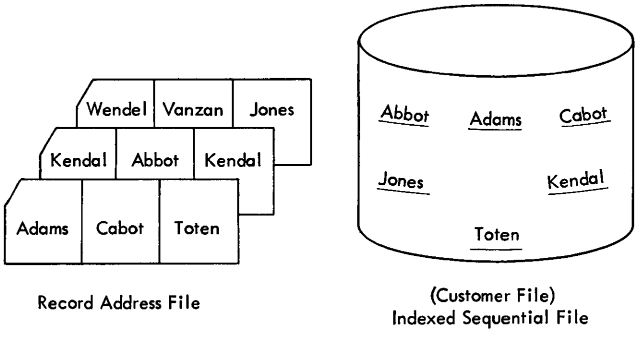


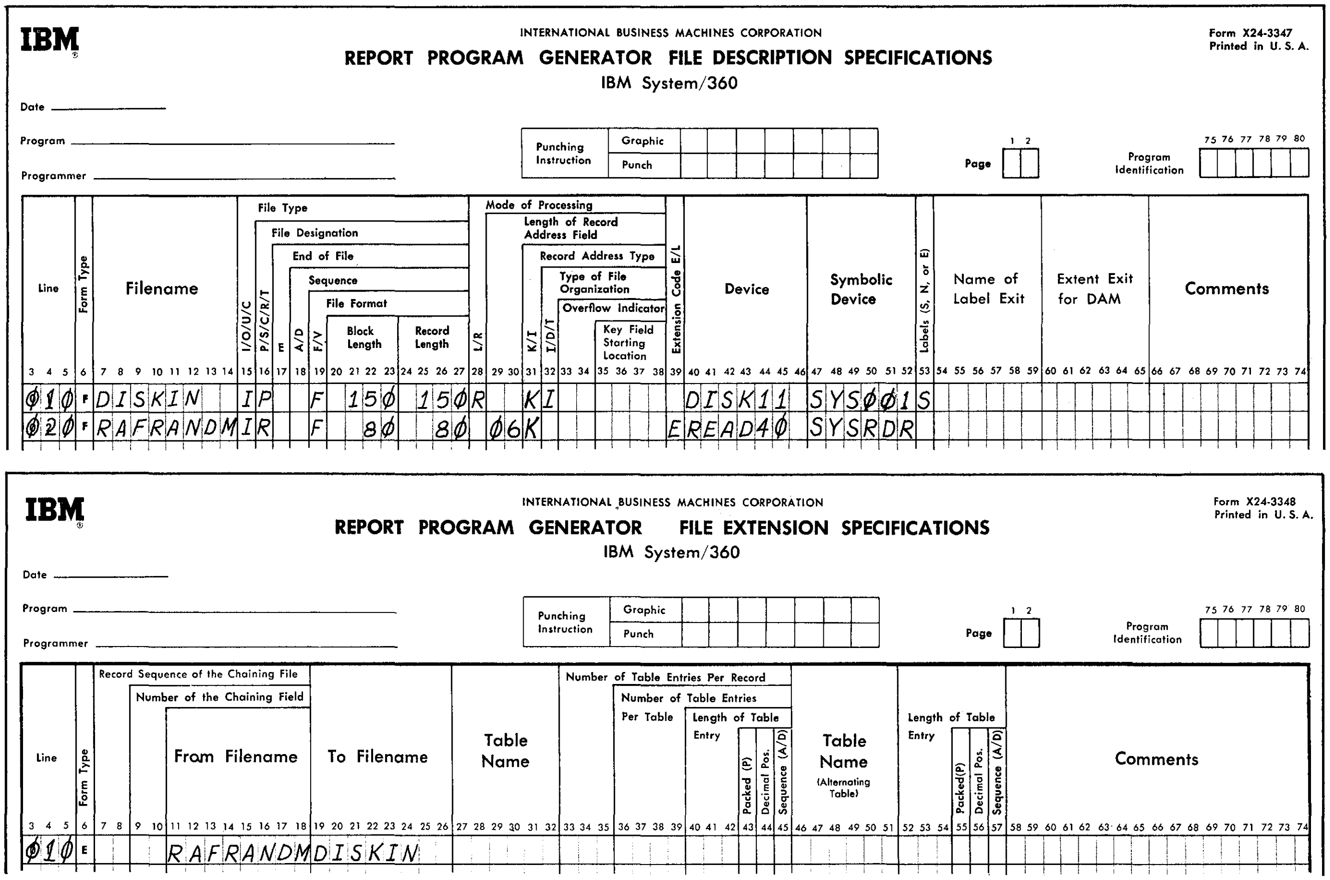
Figure 79. Randomly Processing an Indexed-Sequential File

DIRECT ORGANIZATION

A file with direct organization must reside on a direct access storage device. Records are retrieved from this file by using a track address and either a record key or record identification.

Direct organization is specified by a D in column 32 of the File Description Specification sheet. The mode of processing is random, and an R in column 28 indicates random processing. If the record key is used to obtain the records, enter a K in column 31. If record ID is used, enter an I in column 31.

When an RPG program processes an input file of this organization, an RAF must be supplied. The necessary steps must be supplied which convert the data fields contained in the RAF to the track address and record key or record ID to be retrieved. The conversion routine



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	I/O/U/C (15)	P/S/C/R/T (16)	E (17)	A/D (18)	F/V (19)	Block Length (20-23)	Record Length (24-27)	L/R (28)	Length of Record Address Field (29-30)	K/I (31)	I/D/T (32)	Overflow Indicator (33-34)	Key Field Starting Location (35-38)	Extension Code E/L (39)	Device (40-46)	Symbolic Device (47-52)	Labels (S, N, or E) (53)	Name of Label Exit (54-59)	Extent Exit for DAM (60-65)	Comments (66-74)
Ø1Ø	F	DISKIN	I	P			F	15Ø	15Ø	R		K	I				DISK11	SYSØØ1	S			
Ø2Ø	F	RAFRANDM	I	R			F	8Ø	8Ø		Ø6	K				E	READ4Ø	SYSRDR				

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Form X24-3348
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Record Sequence of the Chaining File (7-8)	Number of the Chaining Field (9-10)	From Filename (11-18)	To Filename (19-26)	Table Name (27-32)	Number of Table Entries Per Record (33-35)	Number of Table Entries Per Table (36-39)	Length of Table Entry (40-42)	Packed (P) (43)	Decimal Pos. (44)	Sequence (A/D) (45)	Table Name (Alternating Table) (46-51)	Length of Table Entry (52-54)	Packed(P) (55)	Decimal Pos. (56)	Sequence (A/D) (57)	Comments (58-74)
Ø1Ø	E			RAFRANDM	DISKIN													

● Figure 80. Processing an Indexed-Sequential File Randomly, Using an RAF File

depends, of course, on the way a data field can be converted to the track address of the record.

The conversion routine is unique, according to the needs of a particular installation, but a conversion routine must be provided. To retrieve records from the file, some data fields must be converted to produce the track address of the record.

Supplying Data to be Converted

One entry in the RAF is supplied for each record to be retrieved. The data supplied by the RAF should be such that the track address and either record key or record identification can be derived. Because the RAF is not described on the input specifications, the entries in the RAF will be made available consecutively in a field called CONTD. This field is always alphameric, and it has the record length as specified in columns 29-30 of the File Description Specification sheet.

Associating a Particular Conversion Routine with RAF

The File Extension Specification sheet entry describing the RAF file contains a field name in columns 27-32. This name is an RPG internal label linking the file extension specification with the corresponding EXTCV operation on the Calculation Specification sheet. It appears there as an entry in Factor 1.

Note: There could be another direct organized file in the same object program, for which a different EXTCV routine would be necessary.

The entry of EXTCV in the operation field indicates the point in the RPG program where the random address conversion routine is to be performed. Factor 2 must contain the name (or label) of the external routine. This external conversion routine must follow the same conventions which are specified for the EXIT routine. See Use of Registers in this publication.

The result field must contain the name (or label) of the field which will contain the track address of the record to be retrieved. This is the result of the conversion. If the record is retrieved by key (specified by a K in column 31 of the File Description Specification sheet), KEYCV has to be entered as the operation code of the next calculation specification.

The result field must contain the name (or label) of the field which will contain the actual record key used to locate the record. The external conversion routine is written (in Assembler language) as an independent routine that must be combined with the generated object program. The name of this routine is the label which appears as Factor 2 of the EXTCV operation on the Calculation Specification sheet.

Conversion Operation Codes

The operation codes which follow are used in conjunction with the conversion routines. If there are several conversion routines in the program, these codes are repeated. However, the conversion routine (or routines) must be written first on the Calculation Specification sheet, preceding all other calculations.

EXTCV. This entry is used to indicate that a conversion routine, which is external to the RPG language, is supplied by the user.

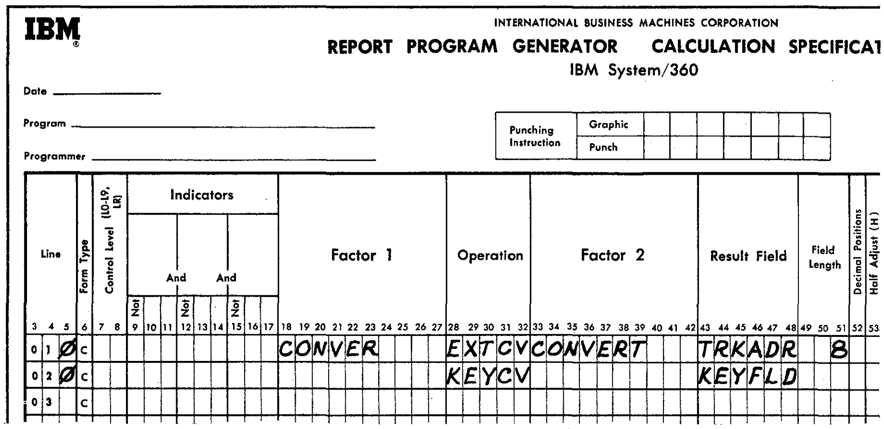
KEYCV. This entry declares that the name specified in the result field will be the field that will contain the record key of the record. This entry must follow the EXTCV entry if the record is retrieved by key.

Figure 81 shows how the calculation specifications are coded for an external conversion routine. In this case, if the records from the disk are retrieved from information in a RAF file, each record address is contained in a special hold area CONTD. The length of this field (CONTD) is specified in columns 29-30 in the File Description Specification sheet. CONTD need not be defined as an RPG Label (RLABL) when EXTCV is used, nor do CONVERT, TRKADR, and KEYFLD.

USING THE ADDROUT (ADDRESS OUTPUT) OPTION

This option could be used to print out a DASD file of records in a sequence other than the sequence in which the file is organized. For example, a payroll file that is organized in sequence by man number and department could be printed out in sequence by social security number.

Instead of sorting the entire file by social security number, the Disk Sort Program merely sets up a new file. This file consists of records containing the addresses of the original data records. This new file has the effect of being a Record Address File. That is, each record in this file directs the program to specific disk addresses to obtain the original data records in social security number sequence.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICAT

IBM System/360

Date

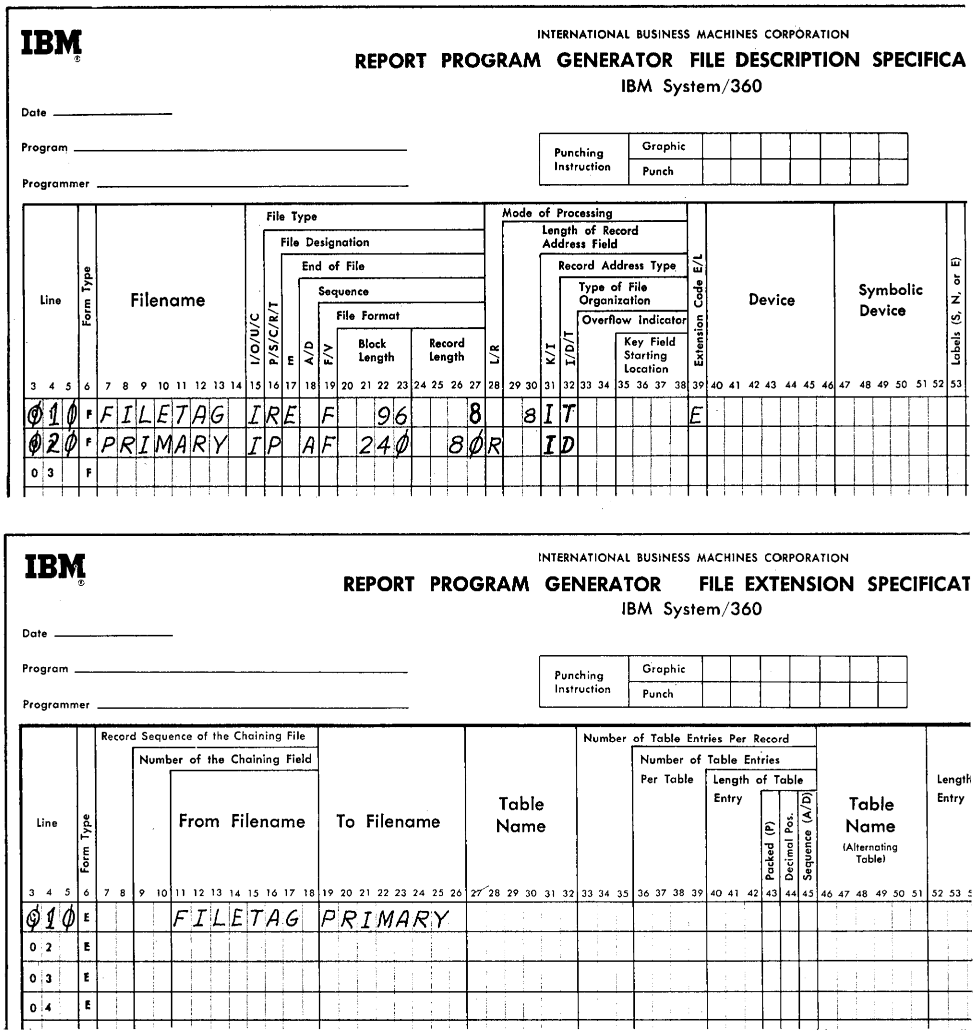
Program

Programmer

Punching Instruction	Graphic							
	Punch							

Line	Form Type	Control Level (L0-L9, LR)	Indicators	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)
3 4 5	6	7 8	9-17	18-27	28-32	33-42	43-48	49-51	52	53
0 1 Ø	C			CONVER	EXTCV	CONVERT	TRKADR	8		
0 2 Ø	C				KEYCV		KEYFLD			
0 3	C									

Figure 81. Specifying a Conversion Routine



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICA

IBM System/360

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Line (3-5)	Form Type (6)	Filename (7-14)	File Type I/O/U/C (15)	File Designation P/S/C/R/T (16)	End of File E (17)	Sequence A/D (18)	File Format F/V (19)	Block Length (20-23)	Record Length (24-27)	Mode of Processing L/R (28)	Length of Record Address Field (29-30)	Record Address Type K/I (31)	Type of File Organization I/D/T (32)	Overflow Indicator (33-34)	Key Field Starting Location (35-38)	Extension Code E/L (39)	Device (40-46)	Symbolic Device (47-52)	Labels (S, N, or E) (53)
Ø1Ø	F	FILETAG	I	R	E		F	96	8		8	I	T			E			
Ø2Ø	F	PRIMARY	I	P		A	F	24Ø	8Ø	R		I	D						
03	F																		

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICAT

IBM System/360

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Line (3-5)	Form Type (6)	Record Sequence of the Chaining File (7-8)	Number of the Chaining Field (9-10)	From Filename (11-18)	To Filename (19-26)	Table Name (27-32)	Number of Table Entries Per Record (33-35)	Number of Table Entries Per Table (36-39)	Length of Table Entry (40-42)	Packed (P) (43)	Decimal Pos. (44)	Sequence (A/D) (45)	Table Name (Alternating Table) (46-51)	Length of Table Entry (52-5)
Ø1Ø	E			FILETAG	PRIMARY									
02	E													
03	E													
04	E													

Figure 82. Using the ADDROUT Option

Figure 82 illustrates the File Description and File Extension Specification sheets for a DASD file containing a data file and an Address Output (ADDROUT) file.

ADDRESS OUTPUT FILE

The file name Filetag is assigned to the ADDROUT file. It is an input file (I in column 15); it is an RAF file (R in column 16) and the end-of-file condition is determined by this file (E in column 17).

The file format is fixed (F in column 19) as a requirement of the Disk Sort Program.

In this example, the block length is 96 and the record length is 8.

The length of the Record Address Field (columns 29-30) is 8. The 8 bytes are a fixed requirement of the Disk Sort Program.

The entry I in column 31 is required for this operation. It applies not to the address output file (FILETAG) itself, but to the file (PRIMARY) that is to be retrieved by the address output file.

The T in Type of File Organization (column 32) indicates that the file is the output of the ADDROUT option of the Disk Sort Program. The E in Extension Code (column 39) means that a File Extension Specification sheet is required.

ORIGINAL DATA FILE

The file name Primary is assigned to the original data record file. It is an input file (I in column 15). It is a primary file (P in column 16), and it is in ascending sequence (A in column 18). The sequence refers to the sequence in which the file is maintained.

In this example, the file format is fixed (F in column 19). The block length is 240 and the record length is 80. The mode of file processing is random (R in column 28). It is random processing because the file is processed in sequence by social security number, but it is organized in sequence by man number and department.

The entry I must appear in column 31 regardless of how the file is organized, since the records are retrieved by record ID.

FILE EXTENSION SPECIFICATIONS

The file extension specifications are used to identify, for the RPG program, the relationship of these two files. The name FILETAG (essentially an RAF) is specified in From Filename. The name PRIMARY is specified in To Filename because this is the file the RAF is linked to.

Note that Table Name (columns 27-32) is left blank. This field would normally contain the label of the EXTCV specification (Factor 1) that links to the external conversion routine, which converts the data in an RAF to an address for a DASD. When the ADDROUT option is used, no data conversion is necessary because the data in FILETAG is already in disk address format.

Multiple input files may be processed using either the matching record technique or chaining. These two concepts are discussed here.

SEQUENTIAL PROCESSING OF MULTIPLE INPUT FILES (MATCHING)

The Matching Field entry, (entered in columns 61-62 on the Input Specification sheet -- see Figure 83) determines which records of a secondary file are to be processed. Both files are processed sequentially. The following rules apply to the matching-field entry.

1. There can be three matching field specification entries (M1, M2, and M3) per record.

2. The locations of the matching fields within a record-type of a file must remain fixed.

3. When there is more than one record-type in an input file, the locations of the matching fields in the various types need not be the same.

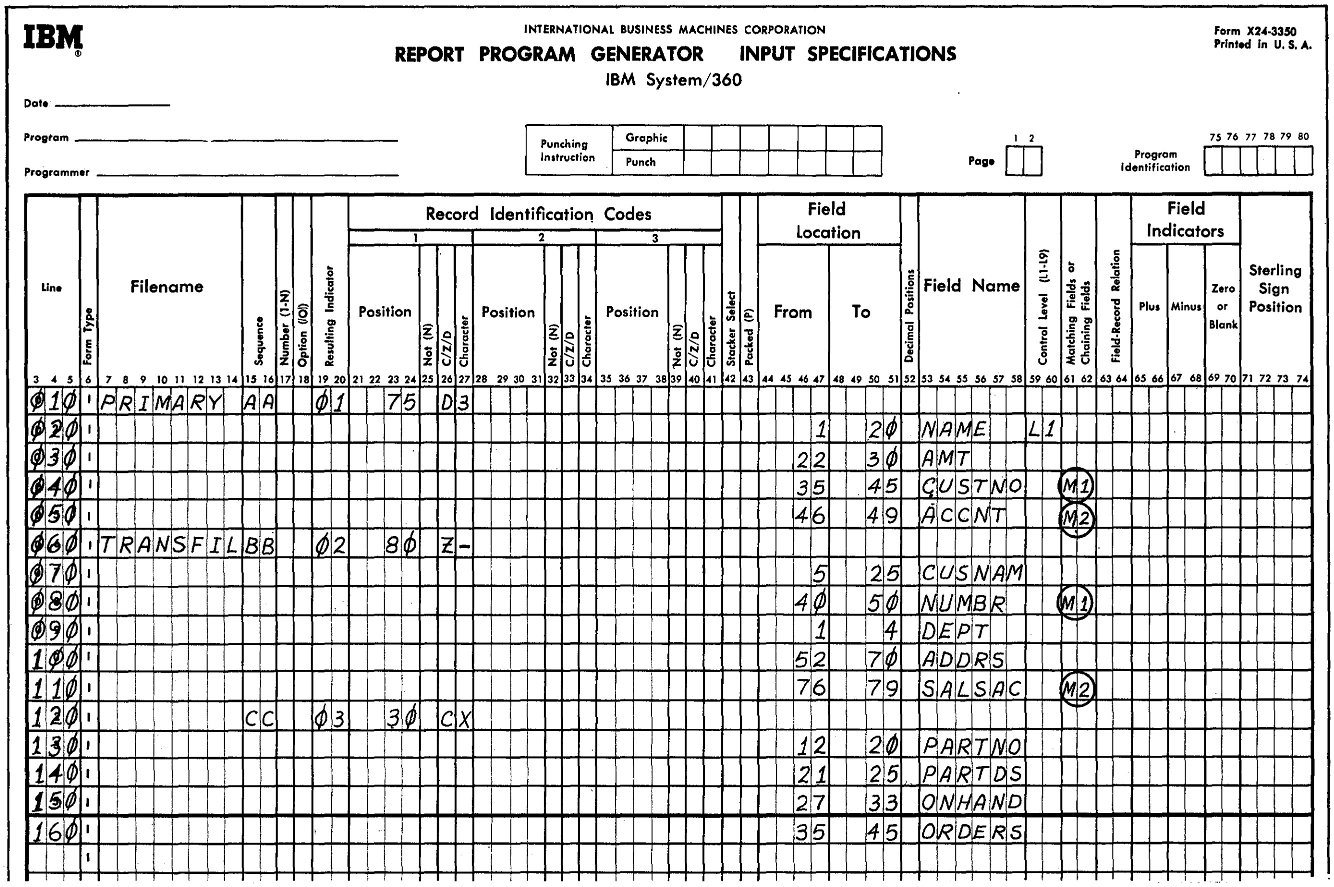
4. Not all the record-types in a file must have a matching field. If an entire file is specified without matching fields, it will be completely read and processed first.

5. If M1, M2, and M3 are specified in the primary file, M1, M2, and M3 must be specified in the secondary files. Incorrect results will be obtained if the same number of matching fields is not specified. For example, Figure 83 shows three record-types. Record-type AA has two matching fields, and record-type BB has two matching fields. If record-type BB had three fields, incorrect results would be obtained. Record type CC has no matching fields (refer to step 4).

The following example may be used to illustrate how the matching field specification is used in conjunction with primary and secondary files. Assume that two files (in sequential organization) are used as shown in Figure 84. The primary file has records which contain heading and rate information. The secondary file contains detailed information which supplements the primary file.

The input specifications required to match these records are shown in Figure 85.

The specifications M1 and M2 cause each detail record to be compared against the primary file's record that has just been read. The fields DIVSON and DETDIV in both files are identified by M2; the field department (DEPT and DETDEP) in both files is identified by M1.



IBM®

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date ________

Program ________

Programmer ________

Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes												Stacker Select	Packed (P)	Field Location		Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators			Sterling Sign Position
							1				2				3																
							Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character			From	To						Plus	Minus	Zero or Blank	
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
Ø1Ø	I	PRIMARY	AA			Ø1	75		D	3																					
Ø2Ø	I																				1	2Ø		NAME	L1						
Ø3Ø	I																				22	3Ø		AMT							
Ø4Ø	I																				35	45		CUSTNO		M1					
Ø5Ø	I																				46	49		ACCNT		M2					
Ø6Ø	I	TRANSFIL	BB			Ø2	8Ø		Z	-																					
Ø7Ø	I																				5	25		CUSNAM							
Ø8Ø	I																				4Ø	5Ø		NUMBR		M1					
Ø9Ø	I																				1	4		DEPT							
1ØØ	I																				52	7Ø		ADDRS							
11Ø	I																				76	79		SALSAC		M2					
12Ø	I		CC			Ø3	3Ø		C	X																					
13Ø	I																				12	2Ø		PARTNO							
14Ø	I																				21	25		PARTDS							
15Ø	I																				27	33		ONHAND							
16Ø	I																				35	45		ORDERS							
	I																														

Figure 83. Matching Field Entries

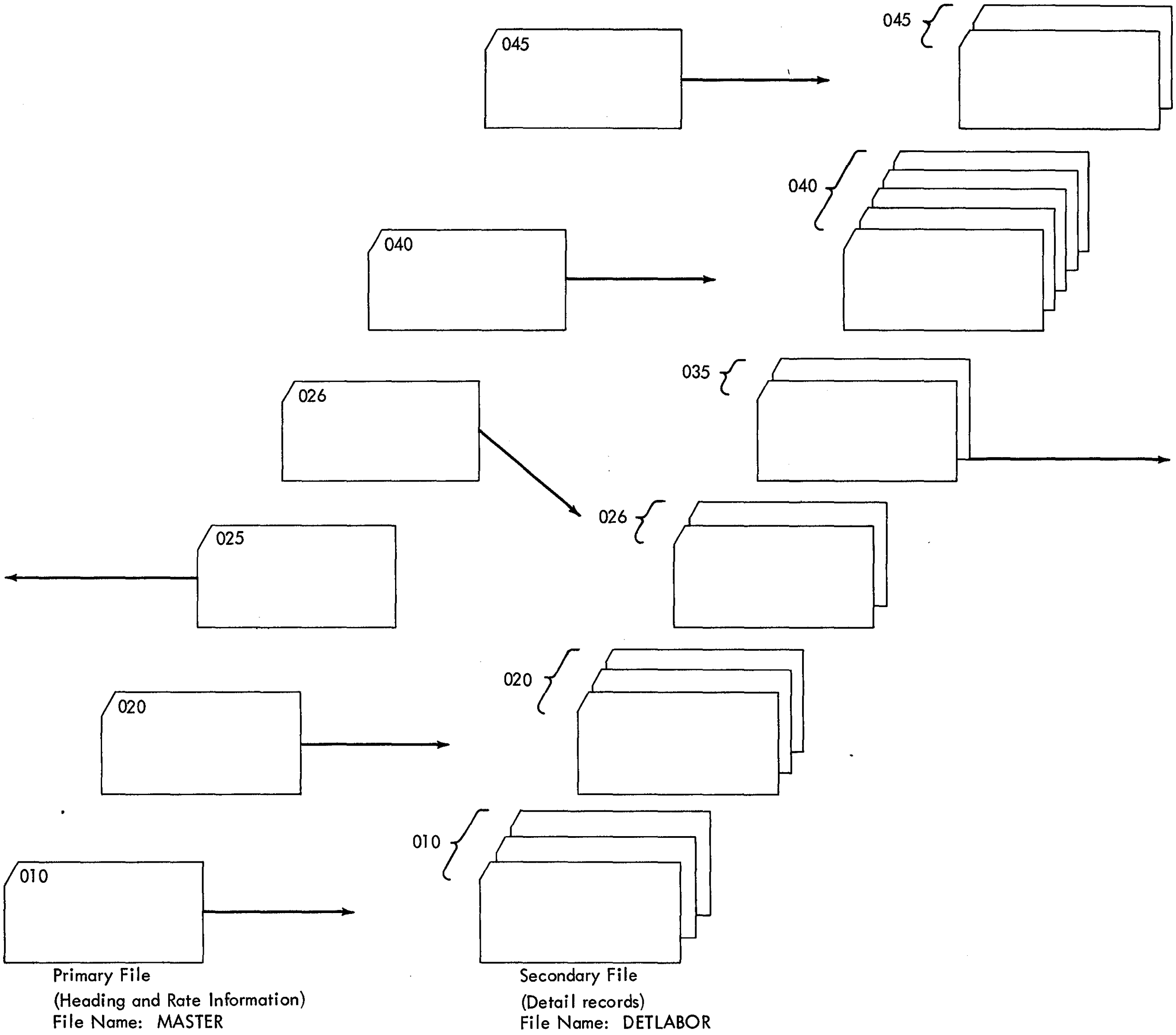
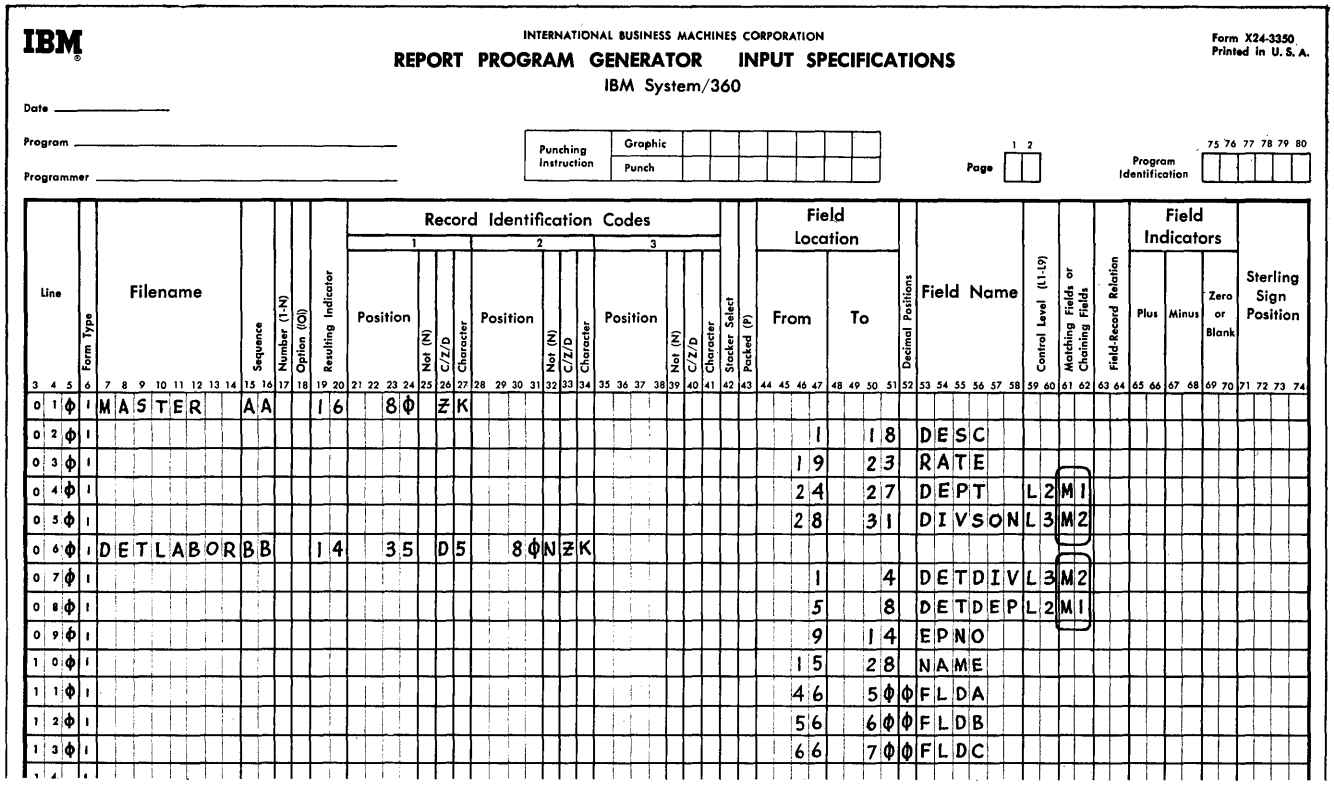


Figure 84. Two Input Files with Matching Records



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	Not (N)	C/Z/D	Character	2: Position	Not (N)	C/Z/D	Character	3: Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	Field Location: From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Minus	Zero or Blank	Sterling Sign Position
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
01Φ	I	MASTER	AA			16	8Φ		Z	K																					
02Φ	I																				1	18		DESC							
03Φ	I																				19	23		RATE							
04Φ	I																				24	27		DEPT	L2	M1					
05Φ	I																				28	31		DIVSON	L3	M2					
06Φ	I	DETLABOR	BB			14	35		D	5	8Φ	N	Z	K																	
07Φ	I																				1	4		DETDIV	L3	M2					
08Φ	I																				5	8		DETDEP	L2	M1					
09Φ	I																				9	14		EPNO							
10Φ	I																				15	28		NAME							
11Φ	I																				46	5Φ	Φ	FLDA							
12Φ	I																				56	6Φ	Φ	FLDB							
13Φ	I																				66	7Φ	Φ	FLDC							
14	I																														

Figure 85. Specifying Matching Fields

MATCHING RECORD INDICATOR

The matching field entries of M1 and M2 (and also M3) have an associated internal indicator MR (Matching Record). This indicator, which is similar to a result indicator, is used to control functions specified on the Calculation and Output-Format Specification sheets.

The MR indicator is turned on when a record of a secondary file matches a record of the primary file. It remains on during the complete processing of the record, and it is turned off when all total calculations and printing that may be caused for this record are completed.

If, as in Figure 84, a detail record does not have a matching primary record (card 035) or if a master record does not have a matching detail record (card 025), the indicator MR is not turned on. This indicator can be used on the calculation specifications to prevent calculations upon a detail record contained in the secondary file. It could also be used on the output specifications to select unmatched detail cards.

The matching-fields specification can be used even though not all of the record-types in the file contain the fields used for matching. When these record-types are specified on the input sheet, the matching fields specification is left blank. This indicates to the program that these record-types do not need to be checked for a matching field. These records are processed immediately after any total operations whose conditions are satisfied.

MATCHING FIELDS SPECIFIED AS NUMERIC OR ALPHAMERIC

Numeric

If a field used as a matching record field has been specified as a numeric field (an entry has been made in Decimal Position), all zone-bits in all positions of the record are removed before the comparison of matching fields is made.

Alphameric

If a field used as a matching record field has been specified as an alphameric field (blank decimal positions), no zone-bits are removed from the record before the comparison is made.

SEQUENCE OF ASSIGNING MATCHING FIELDS

One, two, or three fields can be matched in one operation. However, if more than one field is matched, the designations of M3, M2, and M1 must be assigned in the same sequence in which the fields are to be arranged for matching. M3 is assigned to the highest-order field, M2 to the next lower-order field, and M1 to the lowest order field.

For example, in Figure 85 only two fields are matched. M1 is assigned to DEPT which is the low-order field, and M2 is assigned DIVSON, which is the high-order field.

The position in the record of the fields to be matched does not have to be the same in both files. For example, in Figure 85, the field DETDIV is located in positions 1-4 of the detail records, and the field DIVSON is located in positions 28-31 of the rate-header records.

ORDER OF PROCESSING MATCHED RECORDS

Figure 86 illustrates a primary and a secondary file. The records in the two files will be processed according to four possibilities:

1. Whenever there is a matching primary record
2. Whenever there is a matching secondary record
3. Whenever there is an unmatched primary record
4. Whenever there is an unmatched secondary record.

In Figure 86, indicator 01 is turned on whenever there is a primary record. Indicator 02 is turned on whenever there is a secondary record. Thus,

1. A matching primary record will be coded 01 and MR.
2. A matching secondary record will be coded 02 and MR.
3. An unmatched primary record will be coded 01 and NMR.
4. An unmatched secondary record will be coded 02 and NMR.

The order in which the primary file and the secondary file are processed is listed here. Sample Program Two uses the matching field specifications.

Primary File		Secondary File	
Matching Field in the Record	Processed	Matching Field in the Record	Processed
001	1st	001	3rd
001	2nd	001	4th
002	5th	002	6th
004	9th	003	7th
004	10th	003	8th
006	12th	005	11th

If more than one secondary file is specified, the secondary files will be processed in the sequence they are specified in the File Description Specification sheet.

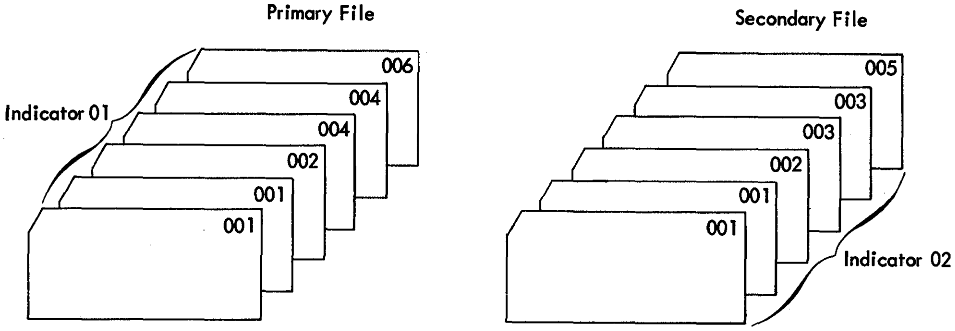


Figure 86. Order of Processing Matched Records

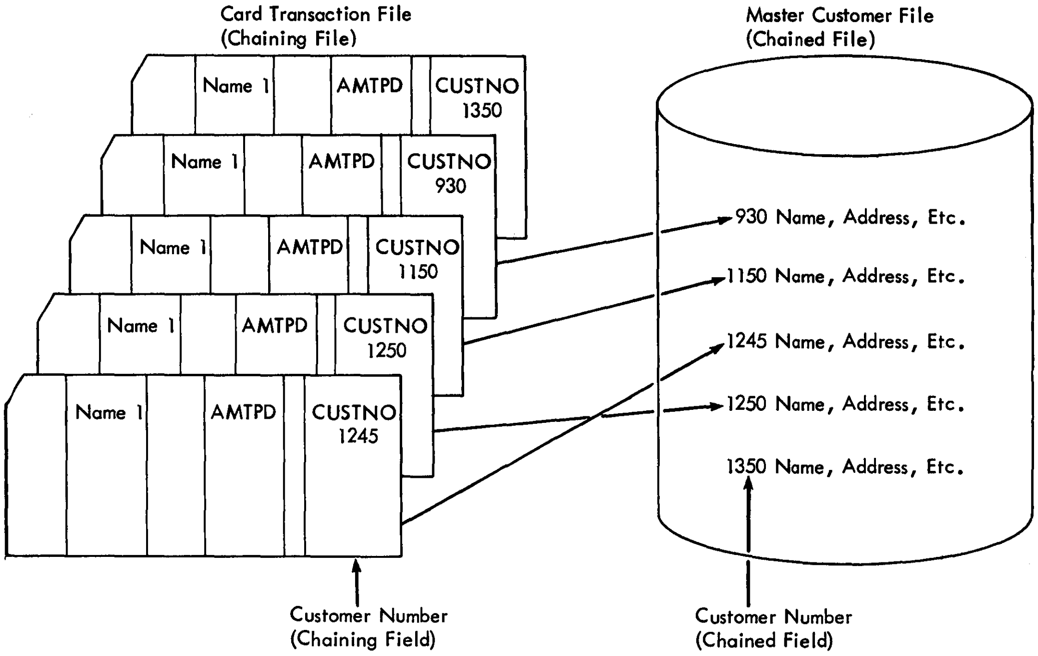


Figure 87. Using Chaining to Process an Indexed-Sequential File

RANDOMLY PROCESSING MULTIPLE INPUT FILES (CHAINING)

To understand chaining, assume that a sequential input file, as shown in Figure 87, contains information about transactions made with several customers. The card file contains the customer's number, but it does not contain his name, address, or balance. Another file called MASTCUST (Master Customer File) contains information about each customer. The RPG object program has the ability to use the second file, when preparing a customer report.

The master customer file has Indexed Sequential Organization, and it is to be processed randomly. The record key in each disk record contains the customers' numbers. The field in the transaction file, which contains the customer's number, can be used to chain the files together. The object program takes the customer's number and locates the record with the same record key. The additional information, such as the customer's name, address, balance, etc., is associated with each record key, and it is immediately available for processing.

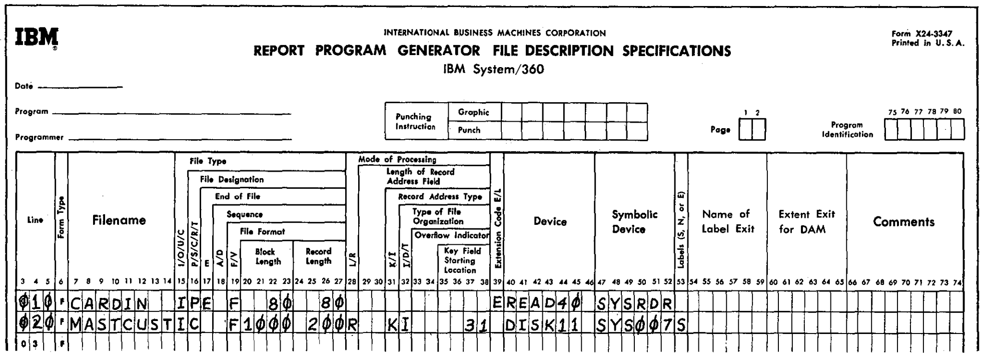
The field which links or chains a record of one file to a record in another file is called a <u>chaining field</u>. The transaction file (CARDIN) is called the <u>chaining file</u> because it links to the chained file. The master customer file (MASTCUST) is the <u>chained file</u> because it is linked to the transaction file.

Up to three chaining fields may be specified for a record. The chaining fields are designated by entering C1, C2, or C3 in columns 61-62 of the Input Specification sheet.

> <u>Note 1</u>: There is no specific relationship between levels C1, C2, or C3 other than specifying the three possibilities for chaining fields.

> <u>Note 2</u>: A chaining field cannot be in packed decimal format and be specified as a numeric field. When packed, the chaining field must be specified as alphameric.

Figure 88 illustrates coding used for chaining the transaction file CARDIN to the customer file MASTCUST. On the File Description Specification sheet, the card file is considered the primary file. When CARDIN is depleted, the program goes to the end-of-job routine (<u>E</u> in column 17).



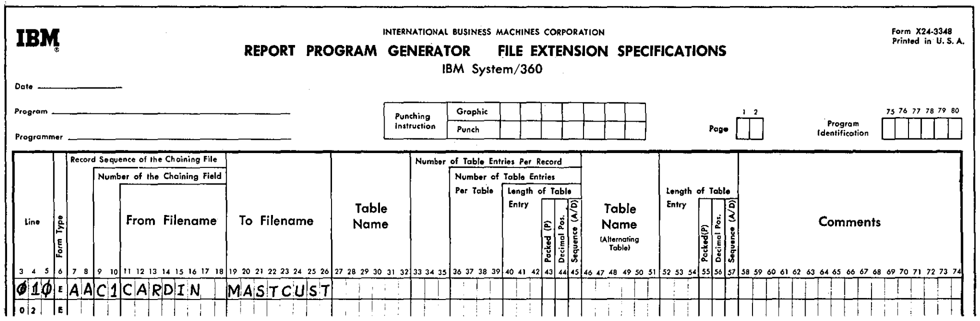
IBM — INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347 Printed in U.S.A.

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	Mode of Processing	Length of Record Address Field	Record Address Type (K/I)	Type of File Organization (I/D/T)	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	Comments
Ø1Ø	F	CARDIN	I	P	E		F	8Ø	8Ø							E	READ4Ø	SYSRDR				
Ø2Ø	F	MASTCUST	I	C			F	1ØØØ	2ØØ	R		K	I		31		DISK11	SYSØØ7	S			
03	F																					



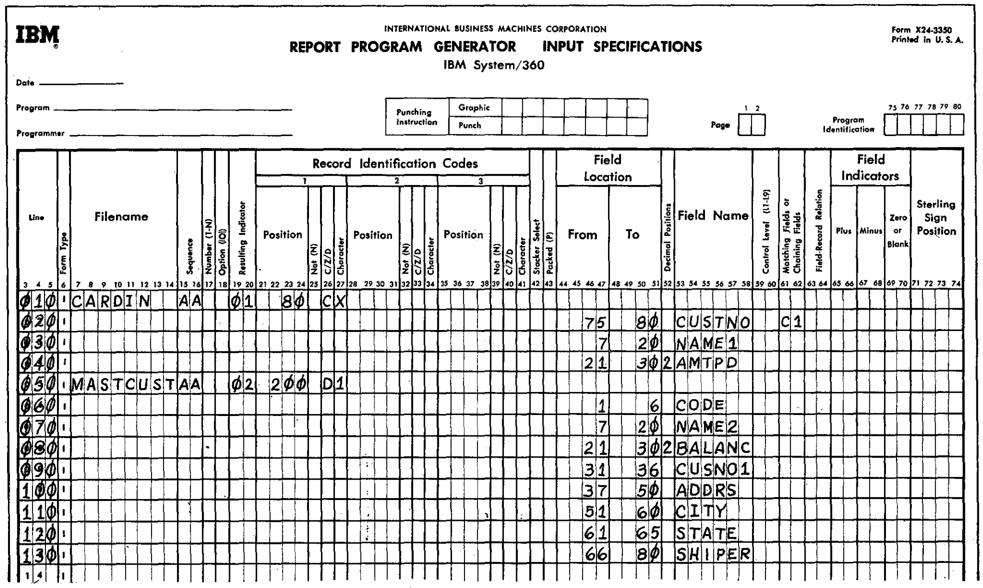
IBM — INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Form X24-3348 Printed in U.S.A.

Line	Form Type	Record Sequence of the Chaining File	Number of the Chaining Field	From Filename	To Filename	Table Name	Number of Table Entries Per Record	Number of Table Entries Per Table	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Table Name (Alternating Table)	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Comments
Ø1Ø	E	AA	C1	CARDIN	MASTCUST													
02	E																	



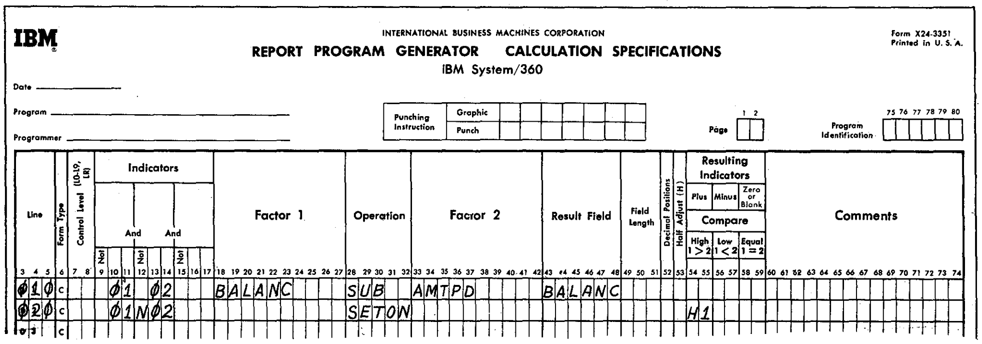
IBM — INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350 Printed in U.S.A.

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1 Position	Not (N)	C/Z/D	Character	From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Plus	Minus	Zero or Blank	Sterling Sign Position
Ø1Ø	I	CARDIN	AA			Ø1	8Ø		C	X											
Ø2Ø	I										75	8Ø		CUSTNO		C1					
Ø3Ø	I										7	2Ø		NAME1							
Ø4Ø	I										21	3Ø	2	AMTPD							
Ø5Ø	I	MASTCUST	AA			Ø2	2ØØ		D	1											
Ø6Ø	I										1	6		CODE							
Ø7Ø	I										7	2Ø		NAME2							
Ø8Ø	I										21	3Ø	2	BALANC							
Ø9Ø	I										31	36		CUSNO1							
1ØØ	I										37	5Ø		ADDRS							
11Ø	I										51	6Ø		CITY							
12Ø	I										61	65		STATE							
13Ø	I										66	8Ø		SHIPER							
14	I																				



IBM — INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351 Printed in U.S.A.

Line	Form Type	Control Level (L0-L9, LR)	Indicators	And	And	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Plus / High 1>2	Minus / Low 1<2	Zero or Blank / Equal 1=2	Comments
Ø1Ø	C		Ø1	Ø2		BALANC	SUB	AMTPD	BALANC							
Ø2Ø	C		Ø1	NØ2			SETON						H1			
03	C															

Figure 88. Specifying Chaining

The File Extension Specification sheet contains the record sequence of the CARDIN file and the number of the chaining field. On the Input Specification sheet, the two files are defined. CUSTNO is the field which chains the files. CUSNO1 is the key field on disk, having a starting location of 31 (as denoted in columns 37-38 of the File Description Specification sheet).

When chaining is used, the following situation may occur. A chaining record (for example, JONES) has no corresponding chained record. (JONES is not present in the chained file.) Such a situation may require special action by the program.

Indicators 01 and 02 are both on when a chaining record has a corresponding chained record. In this case 01 represents the chaining record, and 02 represents the chained record. When 01 and 02 are both on, AMPTD is subtracted from BALANC. However, if there is no corresponding record in the chaining file (01N02), halt indicator H1 is turned on. Thus, the possibility of not having a corresponding record in the chained file is accounted for, and processing will terminate when this situation occurs.

Note: This example illustrates the only time that two resulting indicators (representing two different record types) can be on at the same time.

Figure 89 and 90 show two additional uses of chaining. In Figure 89, the card file contains two fields which are both used as chaining fields (salesman number and customer number). The field containing salesman number is chained to a file that is organized by salesman number. The field containing the customer's number is used to chain to the customer file.

In Figure 90, the card file is chained to the customer file. Within each customer record is a field which may be used to chain to another file. In this case, each record in the customer file contains an account number. The account number is used to chain to the account file.

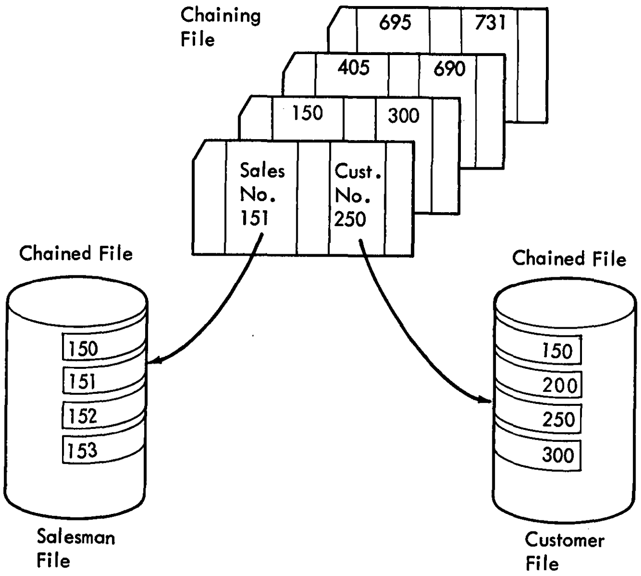


Figure 89. Chaining to Two Files

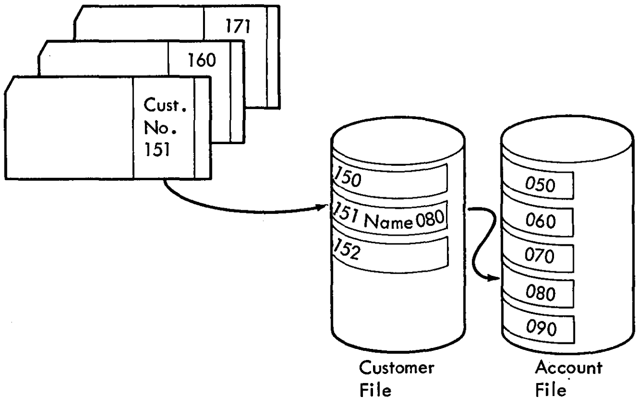


Figure 90. Using a Chained File as a Chaining File

Split Chaining Fields

Several fields that are not in adjoining positions in an input record can be specified as one chaining field. The fields are specified with the same chaining code (C1 for example) on the Input Specification sheet, and the fields are then used as one chaining field. The first field defined on the input sheet is placed in the high-order position, and the last field is placed in the low-order position. The fields are placed in the same sequence as they are defined on the Input Specification sheet.

CREATING RECORD ADDRESS FILES (RAF)

GENERAL INFORMATION

A record-address file is one of the ways by which the necessary information to retrieve records from nonsequential files is supplied to the RPG program. Two types of record-address files may be used:

1. For random processing of a file with Indexed-Sequential organization or a file with direct organization.

2. For sequential processing between limits of a file with Indexed-Sequential organization.

Only one RAF may be specified for an RPG program. An RAF is processed sequentially, and it must be on a file with sequential organization. An RAF is described on the file description sheet and the file extension sheet, but it is not described on the input sheet.

RANDOM PROCESSING OF INDEXED OR DIRECT ORGANIZATION

These rules must be followed when creating a RAF for random processing:

1. For an Indexed Sequential Organization, the field will contain the record key. For a direct organization, each entry in the RAF must consist of a field to be converted to the track address, and either the record Key or the record ID.

2. The record addresses must begin in position 1 of the record.

3. The length of the field must be the same for all records. The numeric fields must always be unpacked.

4. The number of field entries in a record may vary. A blank field, which is equal in length to the record-address field, will cause the RPG program to read the next record in the RAF.

PROCESSING LIMITS OF AN INDEXED SEQUENTIAL ORGANIZATION

When a RAF is used to indicate what limits of an Indexed Sequential Organization are to be processed, the following rules must be observed:

1. Only two record address entries can be in each record.

2. The record address entry must begin in position 1 of the record. The first entry indicates the low limit of the file to be processed. The second entry indicates the upper limit of the file. The program will process from the lower limit to the upper limit.

3. The second entry of the record must begin in the position immediately following the first entry. No blank spaces are allowed.

MULTIPLE FILE PROCESSING

RPG will process multiple files two ways:

1. Sequentially - by using the matching record technique

2. Randomly - by using the chaining technique

Figure 91 shows the processing possibilities for files which have Sequential, Indexed Sequential or Direct Organization.

A sequential organization is processed sequentially and controls the processing of records in:

1. Another Sequential Organization. Both files are processed sequentially, using matching records to govern processing.

2. An Indexed-Sequential Organization. If the file is processed sequentially, the matching-record technique is used to control processing of the Indexed Sequential File. If the file is processed randomly, chaining fields in the sequential file specify which records in the Indexed Sequential Files are to be processed.

To From	Sequential Organization	Indexed Sequential Organization (DASD)	Direct Organization (DASD)
Sequential Organization	Sequential Processing (Matching)	Sequential Processing (Matching) or Random Processing (Chaining)	Random Processing (Chaining)
Indexed-Sequential Organization *	¤ Sequential Processing (Matching)	Sequential Processing ¤ (Matching) or Random Processing (Chaining)	Random Processing (Chaining)
Direct (DASD) Organization +	--	Random Processing (Chaining)	Random Processing (Chaining)

* A Record Address File may be used to supply the limits (in the case of sequential processing) or the actual Record Keys (in the case of random processing).

¤ The From File must be specified as sequential.

+ A Record Address File is converted to supply the record locations on the DASD.

Figure 91. Processing Multiple Input Files

3. A Direct Organization. A direct organization will be processed randomly, under control of the sequential file. The sequential file will contain chaining fields which will be converted to the record locations on the Direct File.

   Note: In all three cases, the Sequential Organization will be processed sequentially.

An Indexed-Sequential Organization may be processed sequentially or randomly, and it controls processing of records in:

1. A Sequential Organization. The records in both files will be processed sequentially, using matching records to control processing.

2. Another Indexed-Sequential Organization. If the file is processed sequentially, matching records are used to control processing. (Both files are processed sequentially.) If the file is processed randomly, chaining fields will be used to control processing.

3. A Direct Organization. Chaining fields in the indexed file are converted to supply the record locations in the direct file which will be processed. The direct file is processed randomly.

A direct organization is processed randomly and controls processing of records in:

1. An Indexed-Sequential Organization. The indexed file is processed randomly, and chaining fields in the direct file control processing of the indexed file.

   In both cases, the direct organization is processed randomly.

2. Another Direct Organization. Chaining fields contained within the direct file are converted to provide the location of the records in another direct organization. The records in both files are processed randomly.

The source deck prepared by the programmer will be arranged as shown in Figure 92. The contents of the basic operating system Job Control statements are listed in the programmer's guide listed on the front cover of this publication.

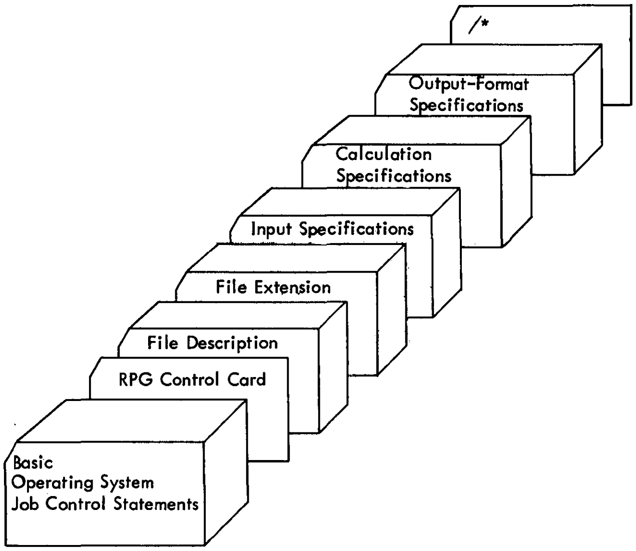

/*
Output-Format Specifications
Calculation Specifications
Input Specifications
File Extension
File Description
RPG Control Card
Basic Operating System Job Control Statements

Figure 92. RPG Source Deck Arrangement in the Basic Operating System Input Stream

The order in which the programmer places his source deck is as follows:

1. Basic Operating System Job Control Statements
2. RPG Control Card (Processor Control Card)
3. File Description Specifications
4. File Extension Specifications
5. Input Specifications
6. Calculation Specifications
7. Output-Format Specifications
8. End-of-File Card

The contents of the RPG control card follow.

Processor Control Card		
Column	Contents	Explanation
1-5	XXXXX	Page and Line numbers.
6	H	Identifies the card as a header card.
7-9	008, 016, 032, 064	Size of Generating Machine (8K, 16K, 32K, or 64K). This entry must not exceed the main storage capacity.
10	P, R, or blank	Enter a P if the object program deck is to be punched in cards (Compile-Only option). Enter an R if the object program is to be written in the relocatable library (permanently) and executed immediately following compilation (Compile-and-Execute option). Leave blank if the object program is to be written in the relocatable library (temporarily) and executed immediately following compilation (Compile-and-Execute option). Note: RPG does not check to determine if the phase name has been previously cataloged in the relocatable library. Therefore, if the user wishes to recatalog the object program at another time, he must first delete the previous entry.
11	D, B, or blank	Type of RPG run. Punch a D if a listing is to be produced and no compilation. Punch a B if a listing is to be bypassed and the compilation is to be performed. (Bypass Listing option). Leave this column blank for both a listing and a compilation.
12-14	008, 016,	Size of Object

	032, 064	Machine (8K, 16K, 32K, or 64K).
15-16	Blank	Leave these columns blank.
17	1, 2, or blank	If the Sterling Shillings Field On Input is in the IBM format, punch 1 in this column. If it is in the BSI format, punch 2. Otherwise leave blank.
18	1, 2, or blank	If the Sterling-Pence Field on Input is in the IBM format, punch 1. If it is in the BSI format, punch 2. Otherwise, leave blank,
19	0, 1, 2, or blank	If the Sterling-Shillings Field on Output is in the printer format, punch 0. If it is in the IBM format, punch 1. If it is in the BSI format, punch 2. Otherwise, leave blank.
20	0, 1, 2, or blank	If the Sterling-Pence Field on Output is in the printer format, punch 0. If it is in the IBM format, punch 1. If it is in the BSI format, punch 2. Otherwise leave blank.
21	I or blank	Inverted Print. If numeric literals and edit words use the European conventions of punctuation (commas for decimal point and vice-versa), enter an I in this column. Otherwise leave blank.
22-25	Blank	Leave these columns blank.
26	A, or blank	Alternate Collating Sequence. Enter an A in this column, if an external subroutine is used to translate the sequence of a matching field to the collating sequence of the System/360. If an external translating subroutine is not used, leave this column blank.
27-74	blank	Leave these columns blank.
75-80	XXXXXX	Program Identification. Enter in these columns the program identification that will appear in the object program identification field. (See Note.)

Note: Object program identification and the sequence number will be punched in each card of the object program deck. If the user specifies the program identification in the RPG Processor control card, XXXXnnnn will be punched in each card (XXXX = the first 4 characters of the program identification; nnnn = sequence number of the card). If no program identification is specified, RPGOnnnn will be punched in each card.

Although only the first 4 characters of the program identification appearing in columns 75-80 will be punched in the object deck, all 6 characters constitute the module name if the object program is cataloged in the relocatable library.

UNBLOCKED RECORDS

The format of a variable-length unblocked record is as follows:

```
┌────┬────┬──────────────┐
│ BL │ RL │ DATA RECORD  │
└────┴────┴──────────────┘
```

where BL is the block length and RL is the record length. Each physical record contains one data record and 2 four-character fields, one containing the block length and one containing the record length. Figure 93 illustrates a segment of a file of variable-length unblocked records (VARUBLK). Data Record 1 contains 53 characters, has a record length of 57, and a block length of 61. However, when specifying these lengths, the user is concerned only with the length of the data record. The four-character record-length field and block-length field are not considered. In the File Description Specification sheet, the data record length of the largest physical record in the file is specified for both record length and block length. In Figure 95, 100 is specified because the largest data record in VARUBLK is Data Record 2, which contains 100 characters. Figure 96 illustrates the coding required in the Output-Format Specification sheet for the variable-length unblocked record file VARUBLK.

BLOCKED RECORDS

The format of a variable-length blocked record is as follows:

```
┌────┬────┬───────────┬────┬───────────┐
│ BL │ RL │DATA RECORD│ RL │DATA RECORD│
└────┴────┴───────────┴────┴───────────┘
```

where BL is the block length and RL is the record length. A physical record contains one or more data records, each having a corresponding record-length field. Figure 94 illustrates one physical record in a file of variable-length blocked records (VARBLK). This physical record contains three data records, having lengths of 53, 100, and 31 characters. (Assume that this physical record is the largest in the file VARBLK.) The total number of characters is 184, which is placed in Block Length (columns 20-23) in the File Description Specification sheet as illustrated in Figure 95. The maximum length of any data record within the file is 100, which appears in Record Length (columns 24-27). If the filename in Figure 96 were changed to VARBLK, the coding illustrated would be equally applicable to the blocked file.

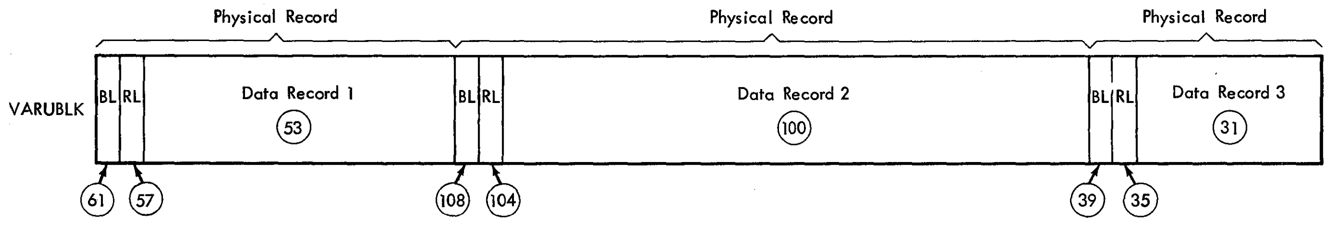


Figure 93. Variable-Length Unblocked Records

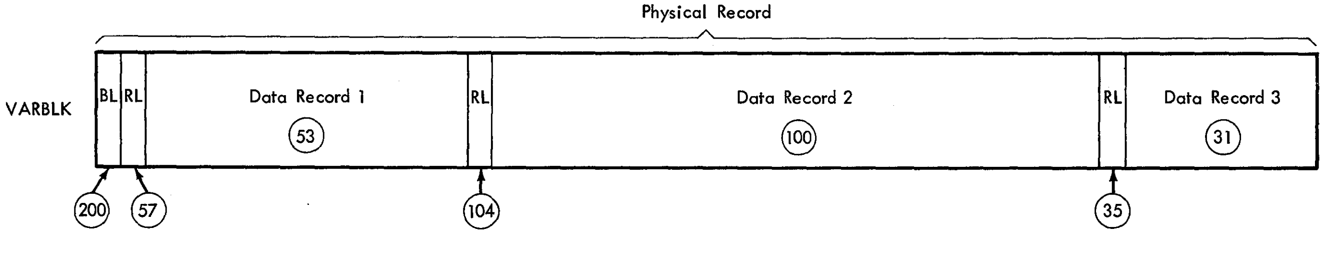
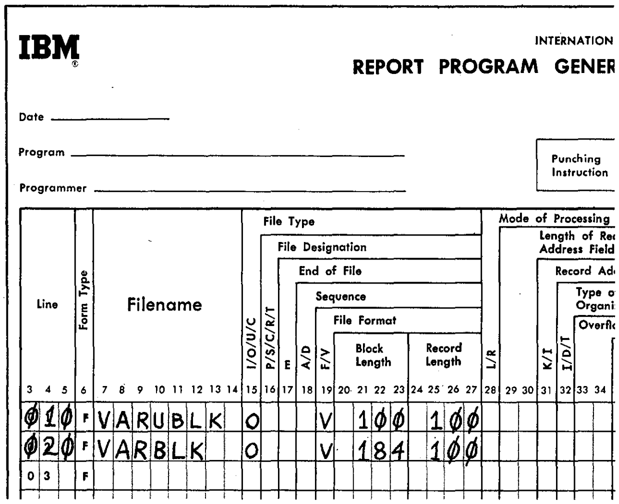


Figure 94. Variable-Length Blocked Records



IBM

INTERNATION

REPORT PROGRAM GENER

Date

Program

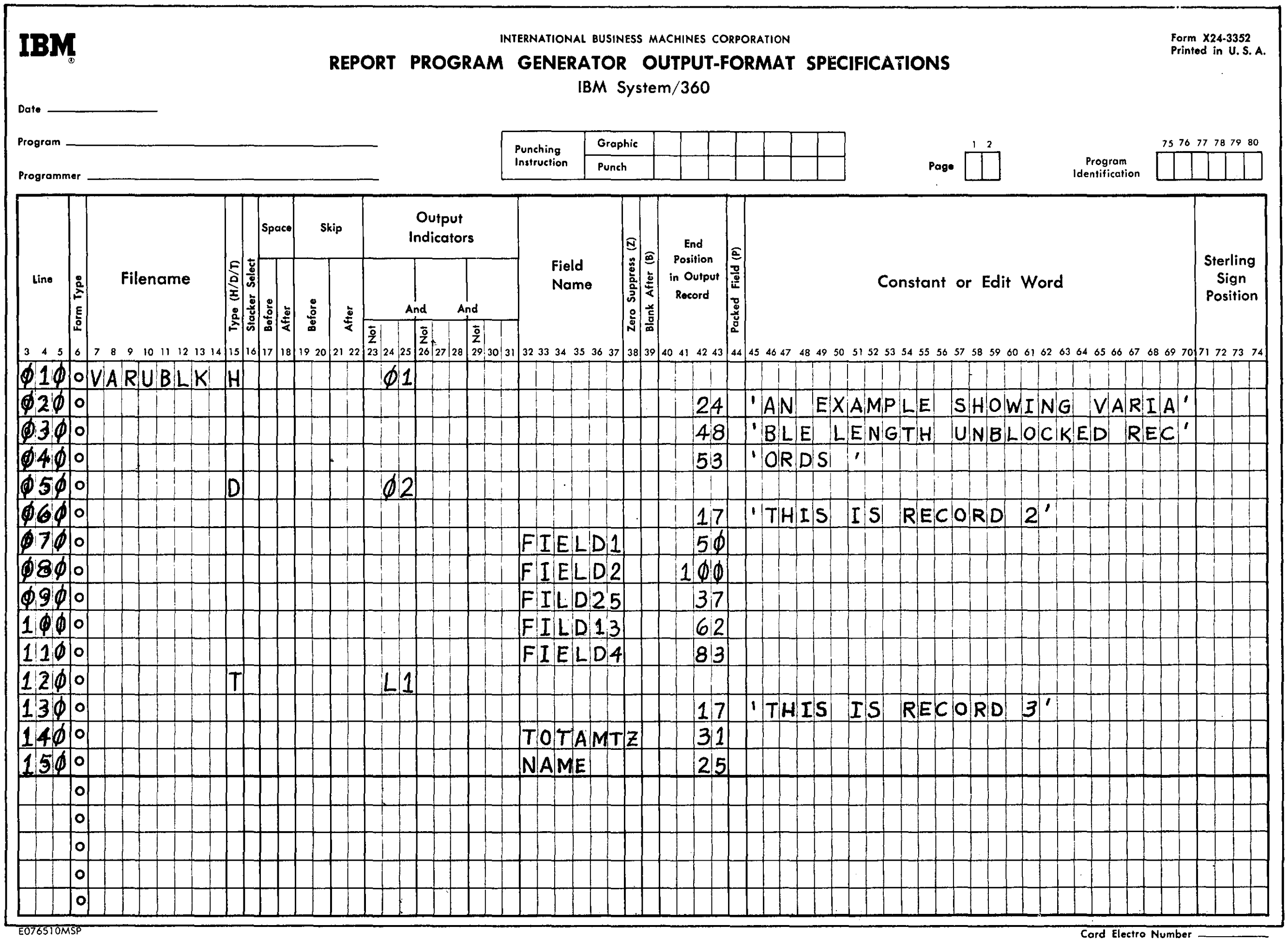
Programmer

Punching Instruction

Line	Form Type	Filename	File Type I/O/U/C	File Designation P/S/C/R/T	End of File E	Sequence A/D	File Format F/V	Block Length	Record Length	L/R	Mode of Processing Length of Re Address Field	Record Ad K/T	Type o Organi I/D/T	Overfl
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
Ø1Ø	F	VARUBLK	O				V	1ØØ	1ØØ					
Ø2Ø	F	VARBLK	O				V	184	1ØØ					
0 3	F													

Figure 95. Example of Block Lengths and Record Lengths of Variable-Length Records

Note: Within RPG, any reference to the record length refers to the length of the data record, and does not include the block-length or the record-length fields. This also applies when specifying the end position in the output record. In columns 40-43 of the Output-Format Specification sheet, the position in the data record is specified without reference to the record-length field.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date ____________

Program ____________

Programmer ____________

Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space		Skip		Output Indicators						Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
					Before	After	Before	After	Not		And Not		And Not								
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
Ø1Ø	o	VARUBLK	H							Ø1											
Ø2Ø	o																	24		'AN EXAMPLE SHOWING VARIA'	
Ø3Ø	o																	48		'BLE LENGTH UNBLOCKED REC'	
Ø4Ø	o																	53		'ORDS '	
Ø5Ø	o		D							Ø2											
Ø6Ø	o																	17		'THIS IS RECORD 2'	
Ø7Ø	o														FIELD1			5Ø			
Ø8Ø	o														FIELD2			1ØØ			
Ø9Ø	o														FILD25			37			
1ØØ	o														FILD13			62			
11Ø	o														FIELD4			83			
12Ø	o		T							L1											
13Ø	o																	17		'THIS IS RECORD 3'	
14Ø	o														TOTAMT	Z		31			
15Ø	o														NAME			25			
	o																				
	o																				
	o																				
	o																				
	o																				

E07651OMSP

Card Electro Number ____________

Figure 96. Output-Format Specification Sheet for a Variable-Length Unblocked Record File

DIAGNOSTICS

The Basic Operating System RPG diagnostic messages to the programmer appear in the following listing. The abbreviated specification types appearing within parentheses (FDS, FES, IS, CS, and OF) refer to the RPG specification sheet on which the erroneous entries were made (File Description, File Extension, Input, Calculation, and Output-Format, respectively).

NOTE	CAUSE OF ERROR	RESULT
1	The File Type entry (FDS, column 15) is not I,O,U, or C, or is blank.	I is assumed.
2	The Mode of Processing entry (FDS, column 28) is invalid for disk device.	Blank is assumed.
3	The Length of Record Address Field entry (FDS, columns 29-30) is invalid or is blank.	08 is assumed.
4	The Record Address Type entry (FDS, column 31) is not K or I.	I is assumed.
5	The File Designation entry of I (FDS, column 16) requires an entry in Extension Code (FDS, column 39).	E is assumed.
6	The File Designation entry (FDS, column 16) is not P,S,R,C, or T for the input file.	P is assumed.
7	The Overflow Indicator entry stated (FDS, columns 33-34) is invalid without a printer device.	Blanks are assumed.
8	The File Type entry stated (FDS, column 15) is invalid with a printer device.	O is assumed.
9	Multiple primary files are stated.	Secondary file is assumed.
10	The Label Indicator entry (FDS, column 53) is not N,S,E, or blank for this tape file.	S is assumed.
11	The Type of File Organization entry (FDS, column 32) is not I,D,T, or blank.	Blank is assumed.
12	The Extension Code entry (FDS, column 39) is not blank for an output file.	Blank is assumed.
13	The End of File entry (FDS, column 17) is not E or blank.	E is assumed.
14	The Sequence entry (FDS, column 18) is not A, D, or blank.	A is assumed.
15	The File Designation entry (FDS, column 16) is not blank for an output file.	Blank is assumed.

NOTE	CAUSE OF ERROR	RESULT
16	The C entry in File Type, (FDS, column 15) requires a card device entry in Device (FDS, columns 40-46).	READ42 is assumed.
17	A Record Address File requires an entry in Extension Code (FDS, column 39).	E is assumed.
18	The File Format entry (FDS, column 19) is not F or V.	V is assumed.
19	The Block Length entry (FDS, columns 20-23) is not a numeric entry or is not right-justified.	The entry in Record Length (FDS, columns 24-27) is assumed.
20	The Record Length entry (FDS, columns 24-27) is invalid or is not right-justified.	0080 is assumed.
21	The U entry in File Type (FDS, column 15) requires a disk device entry in Device (FDS, columns 40-46).	DISK is assumed.
22	An End of File entry (FDS, column 17) and a Sequence entry (FDS, column 18) are invalid with a printer device.	Blanks are assumed.
23	Columns 28-32 (FDS) are not used with an output file.	Blanks are assumed.
24	More than two secondary files are specified.	Additional card(s) not processed.
25	The Overflow Indicator entry (FDS, columns 33-34) invalid.	Blanks are assumed.
26	The Labels entry (FDS, column 53) is incompatible with the device specified.	Blank is assumed.
27	The End of File entry (FDS, column 17) is not blank for an output file.	Blank is assumed.
28	The disk device entry in Device (FDS, columns 40-46) requires an S or E entry in Labels (FDS, column 53).	S is assumed.
29	The Extension Code entry (FDS, column 39) is not E or blank for an input file.	E is assumed
30	The From Filename entry (FES, columns 11-18) is not present or is not left-justified.	Card not processed.
31	The From Filename entry (FES, columns 11-18) is not defined.	Card not processed.
32	The From Filename entry (FES, columns 11-18) is improperly defined.	Card not processed.
33	Number of the Chaining Field entry (FES, columns 9-10) is not C1, C2, or C3.	Card not processed.

NOTE	CAUSE OF ERROR	RESULT
34	The Record Sequence entry (FES, columns 7-8) is invalid. The positions are neither both numeric nor both alphabetic.	Card not processed.
35	The To Filename entry (FES, columns 19-26) is not present or is not left-justified on RAF or Chaining type specifications.	Card not processed.
36	The To Filename entry (FES, columns 19-26) is not defined on RAF or Chaining type specifications.	Card not processed.
37	The To Filename entry (FES, columns 19-26) is improperly defined on RAF or Chaining type specifications.	Card not processed.
38	Columns 33-57 are not all blank on the Chaining type File Extension entry.	Columns 33-57 are ignored.
39	Columns 7-10 are not all blank on the RAF type File Extension entry.	Columns 7-10 are ignored.
40	Columns 33-57 are not all blank on the RAF type File Extension entry.	Columns 33-57 are ignored.
41	Columns 7-10 are not all blank on the Table type File Extension entry.	Columns 7-10 are ignored.
42	The To Filename entry (FES, columns 19-26) is not left-justified on a Table type specification.	Blanks are assumed.
43	The To Filename entry (FES, columns 19-26) is not defined on a Table type specification.	Blanks are assumed.
44	The To Filename entry (FES, columns 19-26) is improperly defined on a Table type specification.	Blanks are assumed.
45	The Table Name entry (FES, columns 27-32 and/or columns 46-51) is not present or is not left-justified.	Card not processed.
46	The letters T A B are not present in columns 27-29 and/or columns 46-48.	Letters T A B are assumed.
47	The Number of Table Entries Per Record entry (FES, columns 33-35) is not numeric.	010 is assumed.
48	The Number of Table Entries Per Table entry (FES, columns 36-39) is not numeric.	0100 is assumed.
49	The Length of Table Entry entry (FES, columns 40-42 and/or columns 52-54) is not numeric.	008 is assumed.
50	The Packed entry (FES, column 43 and/or column 55) is not P or blank.	Blank is assumed.
51	The Decimal Pos. entry (FES, column 44 and/or column 56) is not blank or a numeric digit.	0 is assumed.
52	The Sequence entry (FES, column 45 and/or column 57) is not A, D, or blank.	Blank is assumed.

NOTE	CAUSE OF ERROR	RESULT
53	The Form Type entry (FES, column 6) is not E.	Card not processed.
55	The Name of Label Exit entry (FDS,columns 54-59) is invalid.	Blanks are assumed.
56	Columns 60-65 (FDS) are not all blank.	Blanks are assumed.
57	The Name of Label Exit entry (FDS,columns 54-59) must be blank unless the Labels entry (FDS, column 53) is N or E.	Blanks are assumed.
60	No RPG Processor Control Card.	
61	Column 11 of RPG Control Card does not contain D, B, or blank or is invalid.	Blank is assumed.
62	The machine size (RPG Control Card, columns 7-9 and/or columns 12-14) is not 008, 016, 032, or 064, or is invalid.	008 is assumed.
63	Columns 17-20 of RPG Control Card contain invalid entries.	Blanks are assumed.
64	Column 21 of RPG Control Card does not contain I or blank.	Blank is assumed
65	Column 26 of RPG Control Card does not contain A or blank.	Blank is assumed.
66	Column 10 of the RPG Control Card does not contain P, R, or blank.	P is assumed.
67	Columns 70-80 of the RPG Control Card is blank or invalid.	Program name RPGO is assumed. If R is specified in column 10, blanks are assumed.
71	The Result Field entry (CS, columns 43-48) is required but is not present.	****** in columns 43-48. 015 in columns 49-51, 0 in column 52, and blank in column 53 are assumed.
72	The Result Field entry (CS, columns 43-48) should be blank for this operation.	Blanks are assumed.
73	The Factor 1 entry (CS, columns 18-27) is required but is not present.	Literal 1 is assumed.
74	The Factor 2 entry (CS, columns 33-42) is required but is not present.	Literal 1 is assumed.
75	The Resulting Indicators entry/entries (CS, columns 54-59) is/are invalid.	00 is assumed.
76	The Factor 1 entry (CS, columns 18-27) should be blank for this operation.	Blanks are assumed.
77	The Factor 2 entry (CS, columns 33-42) should be blank for this operation.	Blanks are assumed.

NOTE	CAUSE OF ERROR	RESULT
78	Column 7 of Control Level (CS, columns 7-8) does not contain L or blank.	Blank is assumed.
79	All Detail calculations must precede all Total calculations.	L0 is assumed.
80	The Factor 1 entry (CS, columns 18-27) is not left-justified.	Literal 1 is assumed.
81	The Factor 2 entry (CS, columns 33-42) is not left-justified.	Literal 1 is assumed.
82	The entry in Factor 1 (CS, columns 18-27) is an improperly stated literal or field name.	Literal 1 is assumed.
83	The entry in Factor 2 (CS, columns 33-42) is an improperly stated literal or field name.	Literal 1 is assumed.
84	The Factor 1 entry (CS, columns 18-27) is a field name of more than six characters.	The first six characters are assumed.
85	The Factor 2 entry (CS, columns 33-42) is a field name of more than six characters.	The first six characters are assumed.
86	The Operation entry (CS, columns 28-32) is not present or is invalid.	The MOVE operation code is assumed.
89	The Result Field entry (CS, columns 43-48) is invalid.	****** in columns 43-48 015 in columns 49-51 0 in column 52, and blank in column 53 are assumed.
94	The Field Length entry (CS, columns 49-51) is neither blank nor numeric, or is not right-justified.	015 is assumed.
95	The Decimal Position entry (CS, column 52) is not blank or numeric.	0 is assumed.
96	The Half Adjust entry (CS, column 53) is not H or blank.	H is assumed.
97	A resulting indicator is required for this operation.	00 in columns 58-59 are assumed.
98	The Indicators (CS, columns 9-17) entry(s) is invalid.	00 in columns 10-11 is assumed.
109	Input and/or output specifications are missing or are invalid.	Job is terminated.
110	Form Type (column 6) is not I, C, or O and column 7 does not contain *.	Card not processed.
111	The Filename entry (IS, columns 7-14) is not in the filename table.	Card not processed.
112	The Filename entry (IS, columns 7-14) is not correctly defined on the File Description Specification sheet.	Card not processed.

NOTE	CAUSE OF ERROR	RESULT
113	AND card out of sequence (first in deck, after field name, or invalid file type).	Card not processed.
114	No entries in Record Identification Codes (IS, columns 21-41) in card preceding the AND card.	Card not processed.
115	OR card out of sequence (first in deck, after field name, or invalid file type).	Card not processed.
116	The Filename entry (IS, columns 7-14) is not left-justified.	Card not processed.
117	The entry in Filename (IS columns 7-14) begins with a numeric character.	Card not processed.
118	Entries in Filename (IS, columns 7-14) and in Field Name (IS, columns 53-58) are both present.	The Filename entry is assumed.
119	The Sequence entry (IS, columns 15-16) is blank.	AA is assumed..
120	The sequence is descending and alphameric sequence is found after numeric sequence.	The numeric sequence equal to the previous numeric sequence is assumed.
121	Descending numeric sequence is found.	Sequence equal to previous numeric sequence is assumed.
122	Sequence is numeric and Number entry (IS, column 17) does not contain N or 1.	N is assumed.
123	Sequence is numeric and the Option entry (IS, column 18) is not O or blank.	O is assumed.
124	The input record Resulting Indicator entry (IS, columns 19-20) is blank.	00 is assumed.
125	The Stacker Select entry (IS, column 42) is not 1, 2, or blank.	Blank is assumed.
126	An invalid input file has been specified.	No immediate action taken.
127	The entry(s) in Position (IS, columns 21-24, 28-31, or 35-38) contains a nonnumeric character.	0 is assumed.
128	The Not entry (IS, column 25, 32, or 39) is not N or blank.	N is assumed
129	The C/Z/D entry (IS, columns 26, 33, or 40) is not C, Z, or D.	C is assumed.
130	Field name type specification out of sequence (first in deck, after invalid file name, or AND/OR specifications).	Card not processed.
131	The Field Name entry (IS, columns 53-58) is not left-justified.	Card not processed.
132	The Field Name entry (IS, columns 53-58) begins with a numeric character.	Card not processed.

NOTE	CAUSE OF ERROR	RESULT
133	The From entry (IS, columns 44-47) and/or the To entry (IS, columns 48-51) is blank.	0001 is assumed.
134	The From entry (IS, columns 44-47) and/or the To entry (IS, columns 48-51) contains a non-numeric character.	0 is assumed.
135	The input field length (From and To columns 44-51) is less than 1.	Field length of 1 is assumed.
136	Packed input field length (From and To, columns 44-51) is greater than 8.	Field length of 8 is assumed.
137	The Packed entry (IS, column 43) is not P or blank.	P is assumed.
138	The Decimal Positions entry (IS, column 52) is not numeric or blank.	0 is assumed.
139	The numeric field is greater than 15 bytes in length.	Field length of 15 is assumed.
140	The Control Level entry (IS, columns 59-60) does not contain an L in column 59.	L in column 59 is assumed.
141	The Control Level entry (IS, columns 59-60) does not contain a numeric character in column 60.	1 is column 60 is assumed.
142	Column 61 (IS) does not contain C or M.	M is assumed.
143	The Matching Fields or Chaining Fields entry (IS, column 62) is not numeric.	1 is assumed.
144	The Matching Fields entry (IS, columns 61-62) contains a matching value greater than 3.	03 is assumed.
145	Columns 65-68 (IS) contain numeric indicators for a nonnumeric field.	Indicators are ignored.
146	The alphameric field is greater than 256 bytes in length.	Length of 256 is assumed.
147	A Sterling field is indicated with a decimal position greater than 3.	3 is assumed.
148	A Sterling field is indicated without a decimal position specified.	0 is assumed.
149	Record identification specification is out of order, or no fields are indicated for a given record.	No immediate action taken.
150	No Decimal Position is indicated for a packed input field.	0 is assumed.
151	From, To, and Position entries (IS) contain 0000.	0001 is assumed.
159	A field name beginning with TAB is not a table name.	Card not processed.

NOTE	CAUSE OF ERROR	RESULT
160	The Form Type entry (Column 6) should be O.	Card not processed.
161	The specification type cannot be determined.	Card not processed.
162	The Filename entry (OF, columns 7-14) is invalid, not present, or undefined.	Card not processed.
163	The H/D/T entry (OF, column 15) is out of order.	H is assumed.
164	The Type H/D/T entry (OF, column 15) is not H, D, or T.	H is assumed.
165	The Output Indicators entry (OF, columns 23-31) is not present in an OR-relationship.	00 is assumed.
166	The Output Indicators entry (OF, columns 23-31) is not present in an AND-relationship.	Card not processed.
167	Columns 32-70 (OF) contain entries, but should be blank.	Columns 32-70 are ignored.
168	The Field Name entry (OF, columns 32-37) is not left-justified.	Card not processed.
169	The Output Indicators entry (OF, columns 23-25, 26-28, or 29-31) contain an invalid indicator.	Blanks are assumed.
170	The OR card or the AND card is out of order.	Card not processed.
171	Field type specification, when column 15 is blank, is not preceded by a valid record identification specification.	Card not processed.
172	Columns 7-22 (OF) contain entries, but should be blank.	Columns 7-22 are ignored.
173	Leading and/or closing quote signs in column 45 and 70 (OF) are not present.	No editing is performed.
174	The Zero Suppress entry (OF, column 38) contains an invalid or incorrectly used entry.	Blank is assumed.
175	The Blank After entry (OF, column 39) contains an invalid or incorrectly used entry.	Blank is assumed.
176	The Packed Field entry (OF, column 44) contains an invalid or incorrectly used entry.	Blank is assumed.
177	In the AND-relationship, columns 17-22 (OF) contain entries, but should be blank.	Columns 17-22 are ignored.
178	The End Position in Output Record entry (OF, columns 40-43) is not present, not numeric, or incompatible with literal or edit word.	Card not processed.
179	The Constant or Edit Word entry (OF, columns 45-70) does not contain the required leading and/or closing quote signs.	Card not processed.
180	The numeric Result Field entry (CS, columns 43-48) has a length greater than 15 bytes.	Card not processed.

NOTE	CAUSE OF ERROR	RESULT
181	Incorrect alphameric or numeric field(s) have been specified for this Move Zone operation.	Card not processed.
182	In the Alphameric Compare operation, field lengths greater than 40 must be equal.	Field length of 40 is assumed.
183	The Result Field entry (CS, columns 43-48) contains an undefined field name.	Card not processed.
184	The Field Name entry (IS, columns 53-58, or OF, columns 32-37) contains an unreferenced field name or table name.	No immediate action taken.
185	The Field Name entry (IS, columns 53-58; or OF, columns 32-37) contains a multidefined field name.	No immediate action taken.
186	An arithmetic operation is specified in Operation (CS, columns 28-32) with an alphameric factor in Factor 1 (CS, columns 18-27), Factor 2 (CS, columns 33-42), or Result Field (CS, columns 43-48).	Card not processed.
187	A Compare operation is specified in Operation (CS, columns 28-32) comparing an alphameric field with a numeric field.	Card not processed.
188	The result field length specified in Field Length (CS, columns 49-51) may be too small to contain the true result.	No immediate action taken.
189	The entries in Factor 2 (CS, column 33-42) and Result Field (if specified) must be table names when the LOKUP operation code is used.	Card not processed.
190	EXTCV is not followed by the required operation code (KEYCV) or is not specified properly.	KEYCV card not processed.
191	In the TESTZ operation, a numeric field has been specified as the entry in Result Field (CS, columns 43-48).	Card not processed.
192	The label of the TAG operation and the corresponding GOTO are not in Detail or Total calculations.	No immediate action taken.
193	The Half-Adjust entry (CS, column 53) is incompatible with the result field decimal length.	Half-Adjust entry is ignored.
194	The lengths of Factor 1 and Factor 2 are not equal for the LOKUP operation.	Card not processed.
195	EXTCV operations must be the first to appear in the calculation specifications and no more than 4 are allowed.	Card not processed.
196	The MVR operation is not preceded by DIV operation.	MVR operation is ignored.
197	The DIV operation containing an entry in Half-Adjust (CS, column 53) is followed by an MVR operation.	MVR operation is ignored.

NOTE	CAUSE OF ERROR	RESULT
198	In the LOKUP operation, Factor 1 and Factor 2 are not both alphameric or both numeric fields.	Card not processed.
199	HIGH and LOW resulting indicators are specified for the LOKUP operation.	LOW indicator is ignored.
200	No primary file is stated.	Job is terminated.
201	The Form Type entry does not contain F, E, C, or I.	Card not processed.
202	The Filename entry (FDS, columns 7-14) is invalid.	Card not processed.
203	More than 10 filenames specified.	Additional cards not processed.
204	The Filename entry (FDS, columns 7-14) is not referenced.	No immediate action taken.
205	The Symbolic Device entry (FDS, columns 47-52) is invalid.	Job is terminated.
206	The Device entry (FDS, columns 40-46) is invalid.	Job is terminated.
207	The Filename entry (FDS, columns 7-14) is multi-referenced.	No immediate action taken.
208	The file name is multidefined.	Card not processed.
212	Undefined Resulting Indicator.	No immediate action taken.
213	Unreferenced Resulting Indicator.	No immediate action taken.
214	The maximum number of indicators has been exceeded.	Job is terminated.
219	Undefined field name.	No immediate action taken.
220	Conversion label unreferenced.	No immediate action taken
221	Sum of Matching Field lengths incompatible with other record type(s).	No immediate action taken.
222	Table Name multidefined.	No immediate action taken.
223	Matching field hierarchy violated.	No immediate action taken.
224	Split Chaining Fields improperly specified.	No immediate action taken.
225	Split Control Fields improperly specified.	No immediate action taken.
226	Split Matching Fields are not permitted.	No immediate action taken.
227	The Matching Field lengths are incompatible with other record type(s).	No immediate action taken.
228	The Control Field lengths are incompatible with other record type(s).	No immediate action taken.
229	The Chaining Field lengths are incompatible with other record type(s).	No immediate action taken.

NOTE	CAUSE OF ERROR	RESULT
230	The sum of the Control Field lengths is greater than 256.	No immediate action taken.
231	Field area exceeds record size.	No immediate action taken.
232	Printer file record is blank in columns 17-22 (OF).	Space After of 1 is assumed.
233	The Stacker Select entry (OF, column 16) is invalid.	No immediate action taken.
234	The Space Before entry (OF, column 17) is invalid.	Space Before of 1 is assumed.
235	The Space After entry (OF, column 18) is invalid.	1 is assumed.
236	The Skip Before entry (OF, columns 19-20) is invalid.	Blanks are assumed.
237	The Skip After entry (OF, columns 21-22) is invalid.	Blanks are assumed.
238	The field to be packed is not numeric.	No immediate action taken.
239	The field to be zero-suppressed is not numeric.	No immediate action taken.
240	The Sterling option is requested on a nonnumeric field.	No immediate action taken.
241	Edit word is too small for the field.	No immediate action taken.
242	The field to be edited is not numeric.	No immediate action taken.
243	Both fixed and floating dollar sign have been specified.	No immediate action taken.
244	Both CR and - used for CREDIT.	No immediate action taken.
245	The output specifications are not present or are invalid for this program.	Job is terminated.
246	Page indicators are not numeric.	No immediate action taken.
247	The field length is greater than the field End Position indicates.	No immediate action taken.

Three complete sample programs are included in this section.

SAMPLE PROGRAM ONE

The input file in the first program consists of punched cards. Each card in the file contains a data record that includes from one to eighty characters of information. Each data record represents a purchase made from the reporting firm by a customer. The types of information and the card columns in which each appears are shown in Figure 97. Figure 98 gives the complete listing resulting from the processing of this sample program.

The labels assigned to the fields into which the object program will place the information are as follows:

Field	Label
Customer Name	NAME
Invoice Date - Month	MONTH
Invoice Date - Day	DAY
Invoice Number	INVNO
Customer Number	CUSTNO
Customer Location	STATE
Customer Location	CITY
Invoice Amount	INVAMT

To produce an object program capable of writing the report shown in Figure 98 (Part 3), the programmer must prepare a source program as shown in Figure 99. The entries in the RPG specifications sheet are described here.

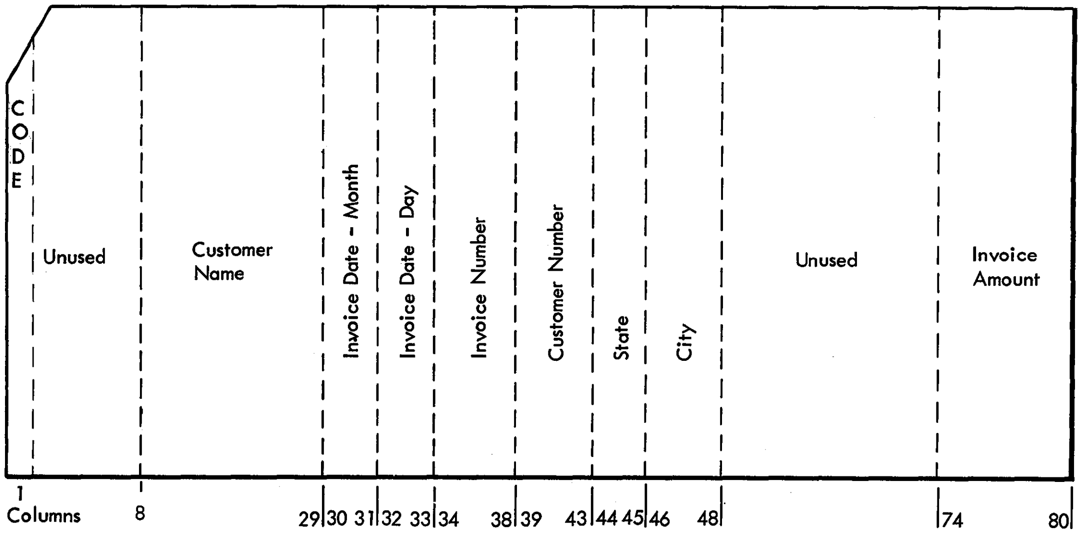
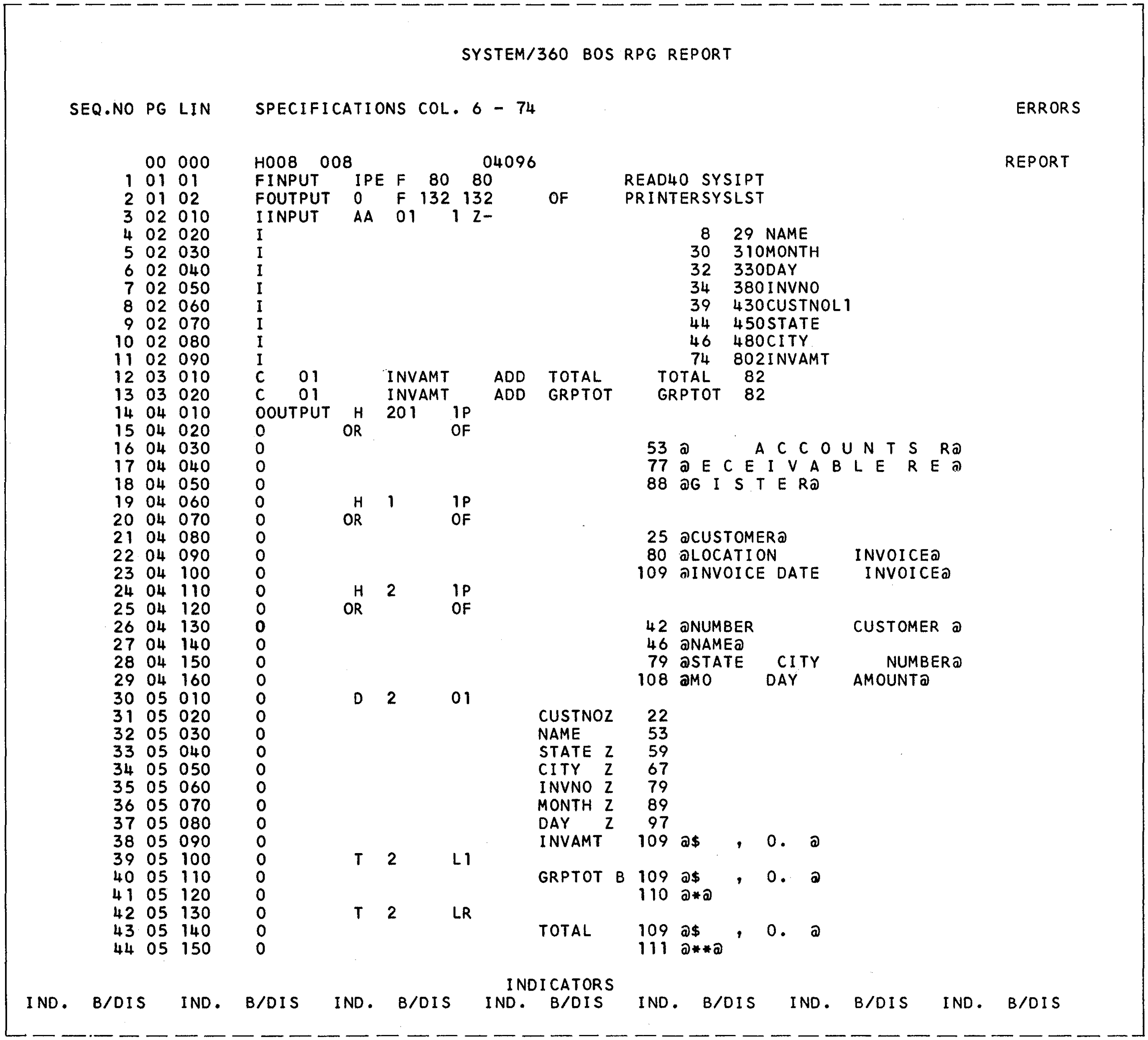
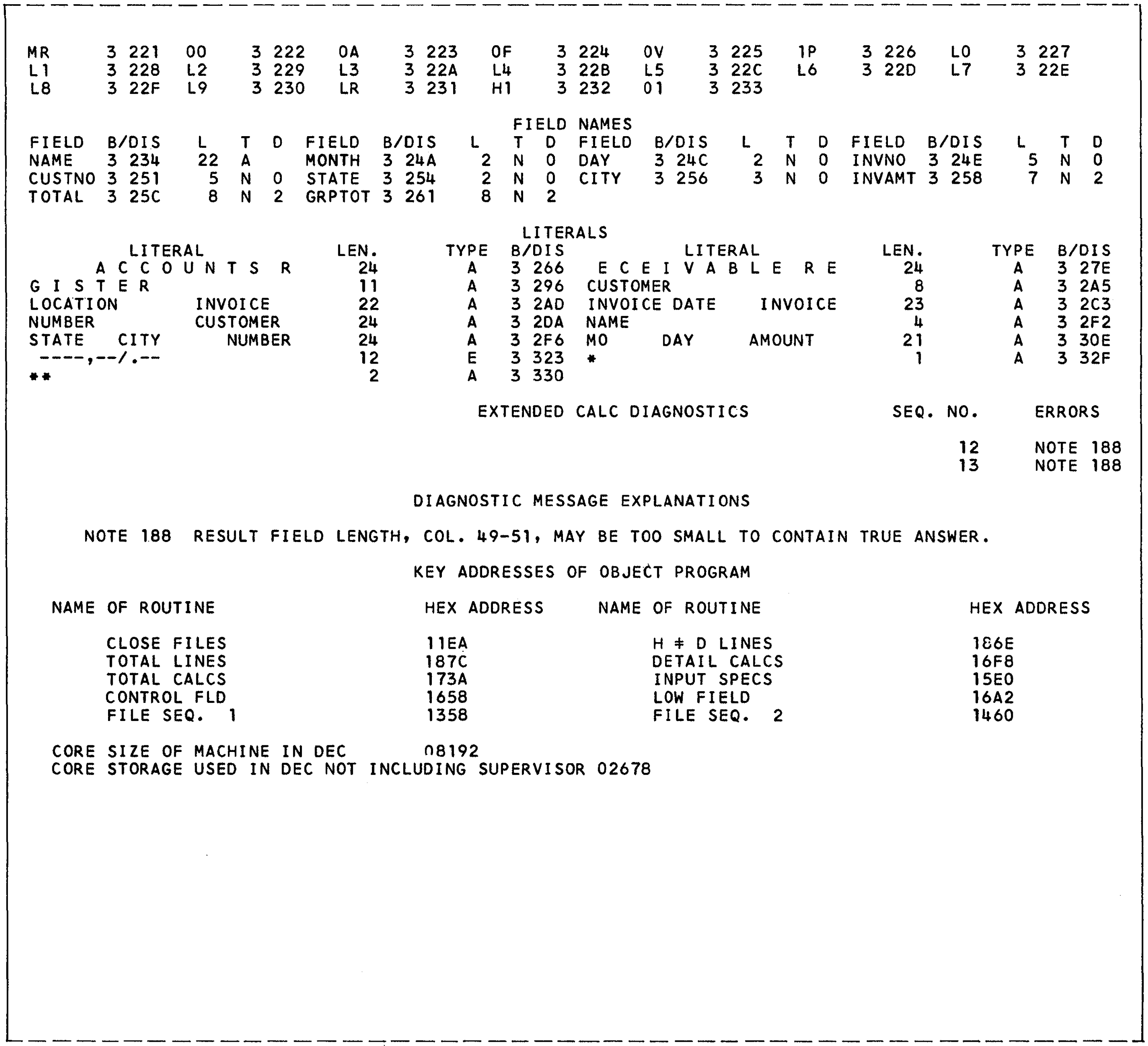


Figure 97. Input File-Card Format



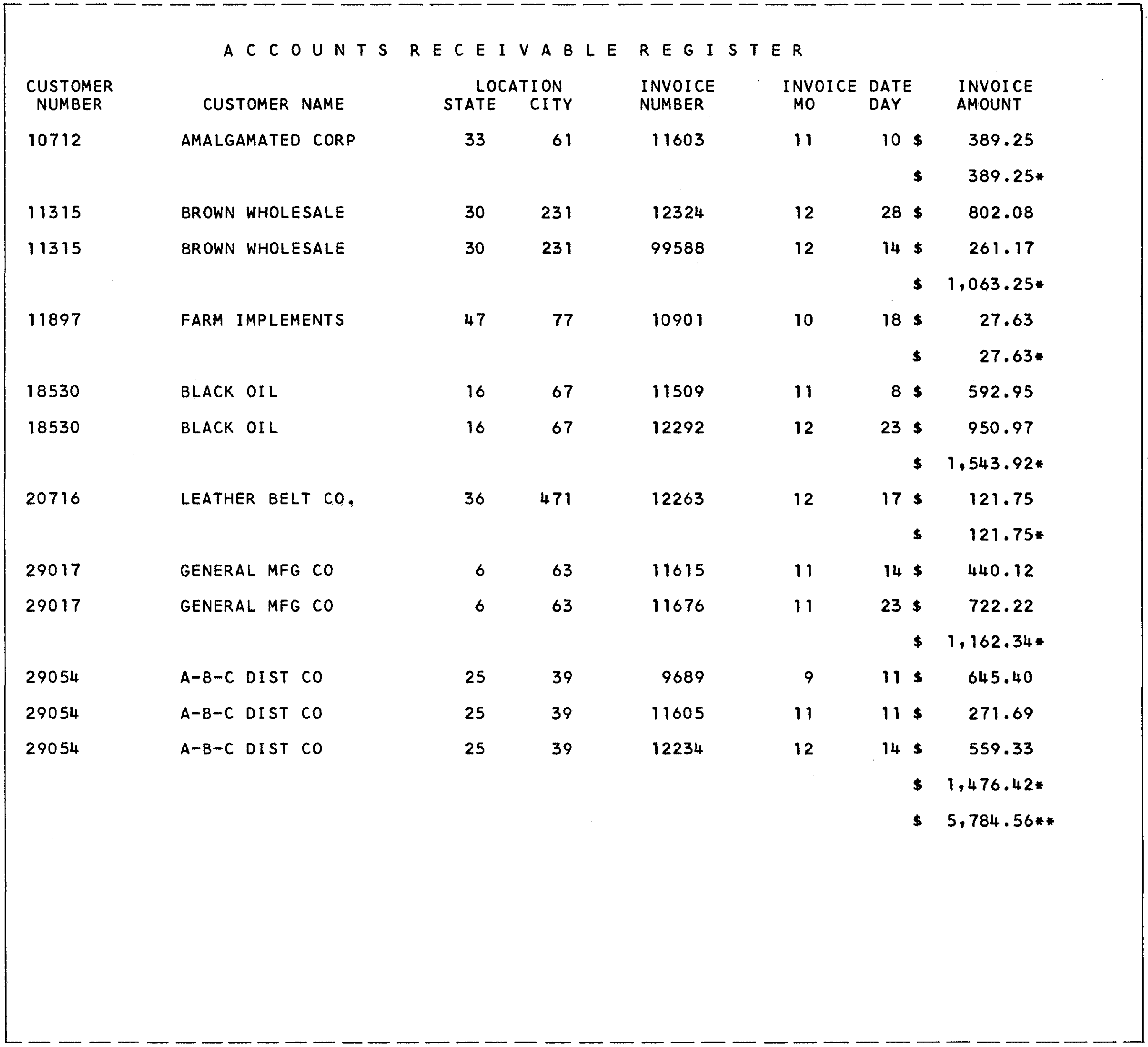
```
                                        SYSTEM/360 BOS RPG REPORT

    SEQ.NO PG LIN    SPECIFICATIONS COL. 6 - 74                                             ERRORS

           00 000    H008  008            04096                                            REPORT
         1 01 01     FINPUT   IPE F  80  80            READ40 SYSIPT
         2 01 02     FOUTPUT  0   F 132 132     OF     PRINTERSYSLST
         3 02 010    IINPUT   AA  01   1 Z-
         4 02 020    I                                        8  29 NAME
         5 02 030    I                                       30  310MONTH
         6 02 040    I                                       32  330DAY
         7 02 050    I                                       34  380INVNO
         8 02 060    I                                       39  430CUSTNOL1
         9 02 070    I                                       44  450STATE
        10 02 080    I                                       46  480CITY
        11 02 090    I                                       74  802INVAMT
        12 03 010    C    01      INVAMT    ADD  TOTAL     TOTAL   82
        13 03 020    C    01      INVAMT    ADD  GRPTOT    GRPTOT  82
        14 04 010    OOUTPUT  H  201   1P
        15 04 020    O       OR        OF
        16 04 030    O                                   53 @      A C C O U N T S  R@
        17 04 040    O                                   77 @ E C E I V A B L E  R E @
        18 04 050    O                                   88 @G I S T E R@
        19 04 060    O        H  1     1P
        20 04 070    O       OR        OF
        21 04 080    O                                   25 @CUSTOMER@
        22 04 090    O                                   80 @LOCATION       INVOICE@
        23 04 100    O                                  109 @INVOICE DATE    INVOICE@
        24 04 110    O        H  2     1P
        25 04 120    O       OR        OF
        26 04 130    O                                   42 @NUMBER         CUSTOMER @
        27 04 140    O                                   46 @NAME@
        28 04 150    O                                   79 @STATE   CITY      NUMBER@
        29 04 160    O                                  108 @MO     DAY     AMOUNT@
        30 05 010    O        D  2     01
        31 05 020    O                         CUSTNOZ   22
        32 05 030    O                         NAME      53
        33 05 040    O                         STATE Z   59
        34 05 050    O                         CITY  Z   67
        35 05 060    O                         INVNO Z   79
        36 05 070    O                         MONTH Z   89
        37 05 080    O                         DAY   Z   97
        38 05 090    O                         INVAMT   109 @$   ,  0.  @
        39 05 100    O        T  2     L1
        40 05 110    O                         GRPTOT B 109 @$   ,  0.  @
        41 05 120    O                                  110 @*@
        42 05 130    O        T  2     LR
        43 05 140    O                         TOTAL    109 @$   ,  0.  @
        44 05 150    O                                  111 @**@

                                            INDICATORS
IND.  B/DIS   IND.  B/DIS   IND.  B/DIS   IND.  B/DIS   IND.  B/DIS   IND.  B/DIS   IND.  B/DIS
```

●Figure 98. Complete Listing of Sample Program One (Part 1 of 3)



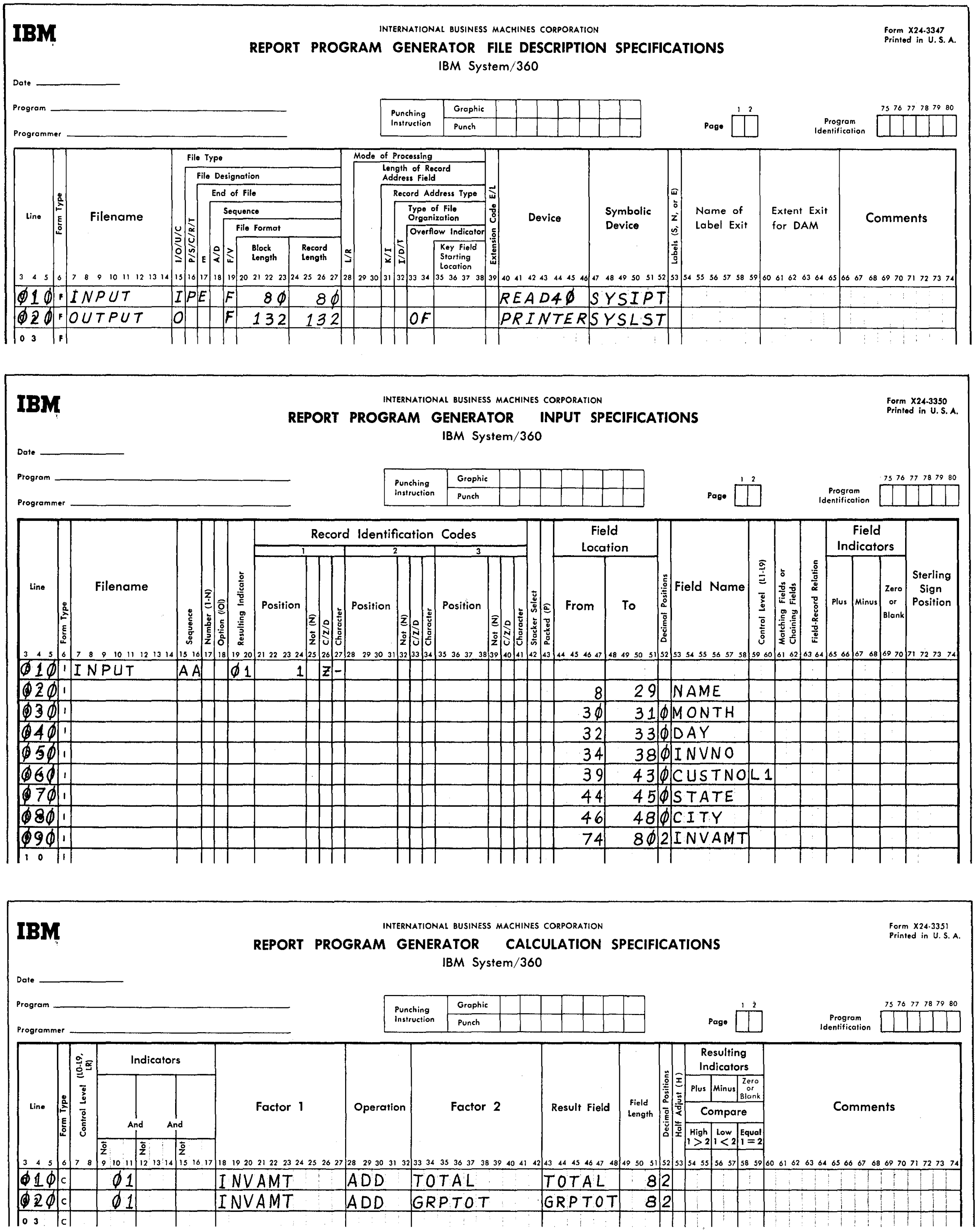
```
MR     3 221  00     3 222  0A     3 223  OF     3 224  0V     3 225  1P     3 226  L0     3 227
L1     3 228  L2     3 229  L3     3 22A  L4     3 22B  L5     3 22C  L6     3 22D  L7     3 22E
L8     3 22F  L9     3 230  LR     3 231  H1     3 232  01     3 233

                                                    FIELD NAMES
FIELD  B/DIS    L   T  D  FIELD  B/DIS    L   T  D  FIELD  B/DIS    L   T  D  FIELD  B/DIS    L   T  D
NAME   3 234   22   A     MONTH  3 24A    2   N  0  DAY    3 24C    2   N  0  INVNO  3 24E    5   N  0
CUSTNO 3 251    5   N  0  STATE  3 254    2   N  0  CITY   3 256    3   N  0  INVAMT 3 258    7   N  2
TOTAL  3 25C    8   N  2  GRPTOT 3 261    8   N  2

                                                     LITERALS
         LITERAL                LEN.      TYPE  B/DIS            LITERAL                LEN.      TYPE  B/DIS
      A C C O U N T S  R         24         A   3 266   E C E I V A B L E  R E           24         A   3 27E
G I S T E R                      11         A   3 296  CUSTOMER                           8         A   3 2A5
LOCATION     INVOICE             22         A   3 2AD  INVOICE DATE     INVOICE          23         A   3 2C3
NUMBER       CUSTOMER            24         A   3 2DA  NAME                               4         A   3 2F2
STATE   CITY      NUMBER         24         A   3 2F6  MO      DAY      AMOUNT           21         A   3 30E
 ----,--/.--                     12         E   3 323  *                                  1         A   3 32F
**                                2         A   3 330

                                                 EXTENDED CALC DIAGNOSTICS             SEQ. NO.     ERRORS

                                                                                            12     NOTE 188
                                                                                            13     NOTE 188

                                           DIAGNOSTIC MESSAGE EXPLANATIONS

     NOTE 188  RESULT FIELD LENGTH, COL. 49-51, MAY BE TOO SMALL TO CONTAIN TRUE ANSWER.

                                           KEY ADDRESSES OF OBJECT PROGRAM

  NAME OF ROUTINE                  HEX ADDRESS     NAME OF ROUTINE                  HEX ADDRESS

       CLOSE FILES                 11EA                 H ǂ D LINES                 186E
       TOTAL LINES                 187C                 DETAIL CALCS                16F8
       TOTAL CALCS                 173A                 INPUT SPECS                 15E0
       CONTROL FLD                 1658                 LOW FIELD                   16A2
       FILE SEQ.  1                1358                 FILE SEQ.  2                1460

  CORE SIZE OF MACHINE IN DEC      08192
  CORE STORAGE USED IN DEC NOT INCLUDING SUPERVISOR 02678
```

● Figure 98. Complete Listing of Sample Program One (Part 2 of 3)



```
                    A C C O U N T S   R E C E I V A B L E   R E G I S T E R

CUSTOMER                                    LOCATION          INVOICE      INVOICE DATE      INVOICE
 NUMBER         CUSTOMER NAME             STATE    CITY       NUMBER         MO     DAY       AMOUNT

10712          AMALGAMATED CORP             33       61        11603         11      10 $     389.25

                                                                                        $     389.25*

11315          BROWN WHOLESALE              30      231        12324         12      28 $     802.08

11315          BROWN WHOLESALE              30      231        99588         12      14 $     261.17

                                                                                        $   1,063.25*

11897          FARM IMPLEMENTS              47       77        10901         10      18 $      27.63

                                                                                        $      27.63*

18530          BLACK OIL                    16       67        11509         11       8 $     592.95

18530          BLACK OIL                    16       67        12292         12      23 $     950.97

                                                                                        $   1,543.92*

20716          LEATHER BELT CO.             36      471        12263         12      17 $     121.75

                                                                                        $     121.75*

29017          GENERAL MFG CO                6       63        11615         11      14 $     440.12

29017          GENERAL MFG CO                6       63        11676         11      23 $     722.22

                                                                                        $   1,162.34*

29054          A-B-C DIST CO                25       39         9689          9      11 $     645.40

29054          A-B-C DIST CO                25       39        11605         11      11 $     271.69

29054          A-B-C DIST CO                25       39        12234         12      14 $     559.33

                                                                                        $   1,476.42*

                                                                                        $   5,784.56**
```

Figure 98. Complete Listing of Sample Program One (Part 3 of 3)



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date ______

Program ______

Programmer ______

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	File Type I/O/U/C	File Designation P/S/C/R/T	End of File E	Sequence A/D	File Format F/V	Block Length	Record Length	L/R	Length of Record Address Field	Record Address Type K/I	Type of File Organization I/D/T	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	Comments
Ø1Ø	F	INPUT	I	P	E		F	8Ø	8Ø								READ4Ø	SYSIPT				
Ø2Ø	F	OUTPUT	O				F	132	132					OF			PRINTER	SYSLST				
0 3	F																					

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date ______

Program ______

Programmer ______

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	Not (N)	C/Z/D	Character	2: Position	Not (N)	C/Z/D	Character	3: Position	Not (N)	C/Z/D	Character	Stacker Select	Packed (P)	Field Location From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators Plus	Minus	Zero or Blank	Sterling Sign Position
Ø1Ø	I	INPUT	AA			Ø1	1		Z	-																					
Ø2Ø	I																				8	29		NAME							
Ø3Ø	I																				3Ø	31	Ø	MONTH							
Ø4Ø	I																				32	33	Ø	DAY							
Ø5Ø	I																				34	38	Ø	INVNO							
Ø6Ø	I																				39	43	Ø	CUSTNO	L1						
Ø7Ø	I																				44	45	Ø	STATE							
Ø8Ø	I																				46	48	Ø	CITY							
Ø9Ø	I																				74	8Ø	2	INVAMT							
1 0	I																														

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date ______

Program ______

Programmer ______

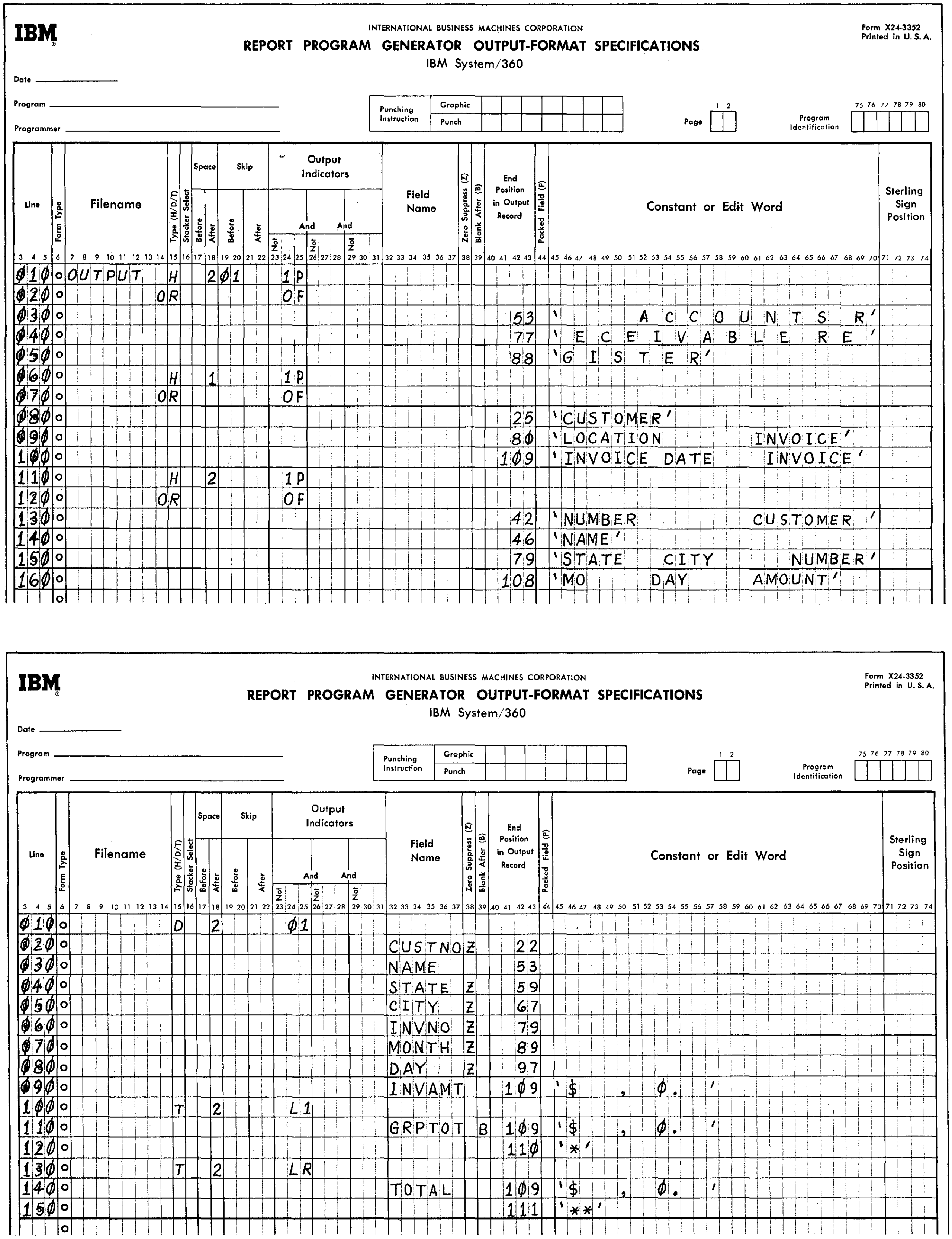
Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Control Level (L0-L9, LR)	Indicators Not	Indicators	And Not	And	And Not	And	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators Plus (High 1 > 2)	Minus (Low 1 < 2)	Zero or Blank (Equal 1 = 2)	Comments
Ø1Ø	C			Ø1					INVAMT	ADD	TOTAL	TOTAL	8	2					
Ø2Ø	C			Ø1					INVAMT	ADD	GRPTOT	GRPTOT	8	2					
0 3	C																		

Figure 99. RPG Specification Sheets--Program One (Part 1 of 2)



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic, Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators: Not (23)	Indicator (24-25)	And: Not (26)	Indicator (27-28)	And: Not (29)	Indicator (30-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)	End Position in Output Record (40-43)	Packed Field (P) (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
Ø1Ø	O	OUTPUT	H			2	Ø1			1P											
Ø2Ø	O	OR								OF											
Ø3Ø	O																	53		' A C C O U N T S R'	
Ø4Ø	O																	77		' E C E I V A B L E R E '	
Ø5Ø	O																	88		'G I S T E R'	
Ø6Ø	O		H			1				1P											
Ø7Ø	O	OR								OF											
Ø8Ø	O																	25		'CUSTOMER'	
Ø9Ø	O																	8Ø		'LOCATION INVOICE'	
1ØØ	O																	1Ø9		'INVOICE DATE INVOICE'	
11Ø	O		H			2				1P											
12Ø	O	OR								OF											
13Ø	O																	42		'NUMBER CUSTOMER '	
14Ø	O																	46		'NAME'	
15Ø	O																	79		'STATE CITY NUMBER'	
16Ø	O																	108		'MO DAY AMOUNT'	
	O																				

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic, Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators: Not (23)	Indicator (24-25)	And: Not (26)	Indicator (27-28)	And: Not (29)	Indicator (30-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)	End Position in Output Record (40-43)	Packed Field (P) (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
Ø1Ø	O		D			2				Ø1											
Ø2Ø	O														CUSTNO	Z		22			
Ø3Ø	O														NAME			53			
Ø4Ø	O														STATE	Z		59			
Ø5Ø	O														CITY	Z		67			
Ø6Ø	O														INVNO	Z		79			
Ø7Ø	O														MONTH	Z		89			
Ø8Ø	O														DAY	Z		97			
Ø9Ø	O														INVAMT			1Ø9		'$, Ø. '	
1ØØ	O		T			2				L1											
11Ø	O														GRPTOT		B	1Ø9		'$, Ø. '	
12Ø	O																	11Ø		'*'	
13Ø	O		T			2				LR											
14Ø	O														TOTAL			1Ø9		'$, Ø. '	
15Ø	O																	111		'**'	
	O																				

Figure 99. RPG Specification Sheets--Program One (Part 2 of 2)

FILE DESCRIPTION SPECIFICATION SHEET

Two files (Input and Output) are described on this sheet. The input file INPUT (columns 7-11) is the primary file (column 16). It causes the end-of-job condition when it is depleted (column 17). The input records are fixed in length (column 19); the block length is 80 (columns 22-23); and the records are 80 characters in length (columns 26-27).

The output file OUTPUT is also defined on the file description sheet. The format is fixed length; the block length is 132 and the records are 132 characters in length. The entry OF in columns 33-34 indicate that the Output File defined on the line is to cause the overflow condition.

INPUT SPECIFICATION SHEET

The input file has a sequence of AA, and if column 1 contains the zone of a minus, resulting indicator 01 is turned on(as indicated by the entries in 19-20, 24, and 26-27). The locations of the fields which contain the input data are defined in columns 46-47. The names of the input fields are entered in columns 53-58. Whenever a new customer is read in, control level 1 is turned on (columns 59-10).

CALCULATION SPECIFICATION SHEET

The contents of the field TOTAL are added to the contents of the field INVAMT, and the result is stored in TOTAL. The result field has a length of eight positions, and two positions are reserved for the decimal portion. The field GRPTOT is added to INVAMT, and the result is stored in GRPTOT which is eight positions long, and has two decimal positions.

OUTPUT-FORMAT SPECIFICATION SHEET

OUTPUT is the name of the file to which the records defined on the line belong, and OUTPUT is entered under Filename on the first line of the sheet. In column 15 the output types H, D, and T are entered to designate the heading, detail, and total lines.

The first heading line, ACCOUNTS RECEIVABLE REGISTER, prints either on the first page (1P) or overflow conditions (OF). The OR entered in columns 14 and 15 of the second line allows for printing on the first page or overflow. The other heading lines are to print on these conditions.

When output indicator 01 is on, the field entered in Field Name will print in the positions indicated in columns 40-43. Zero suppression occurs on CUSTNO, STATE, CITY, INVNO, MONTH, and DAY.

The total lines are to print whenever control fields L1 or LR are on. The group total GRPTOT prints when L1 is on, and after it is printed, the contents of GRPTOT are blanked out. The final total is printed when the LR (last record) indicator is on.

SAMPLE PROGRAM TWO

This is similar to the previous program. In this example, however, two input files are used. The Transaction File is a card file with fields as shown in Figure 96 of the previous program. Another input file (Master Customer File), which is on tape, contains information about the firm's customers (Figure 100). The fields contained in the two input files follow:

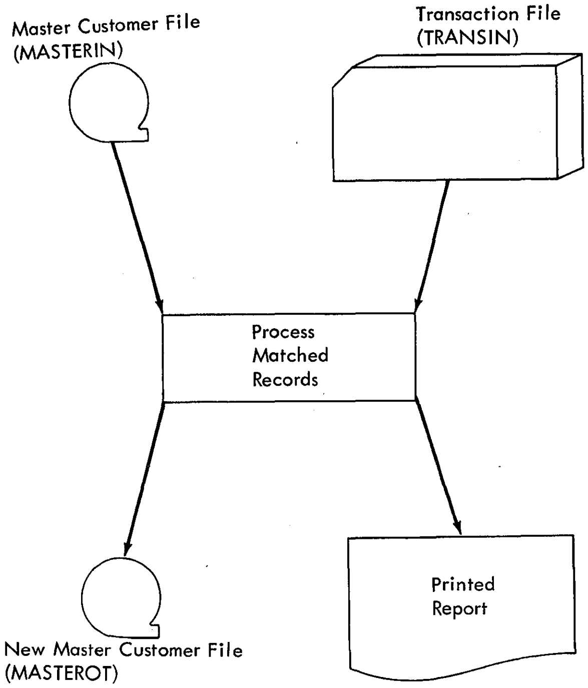


Figure 100. Sample Program Two

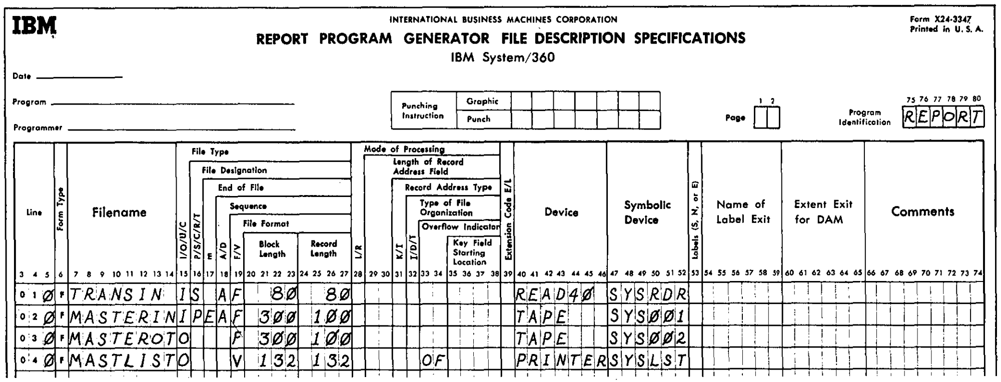
Transaction File

Field	Label	Card Columns
Code	Minus (-) Zone, or Plus (+) Zone	1
Customer name	NAME	8-29
Invoice date	MONTH	30-31
Invoice date	DAY	32-33
Invoice number	INVNO	34-38
Customer number	CUSTNO	39-43
State	STATE	44-45
City	CITY	46-48
Invoice amount	INVAMT	74-80

Master Customer File

Field	Label	Card Columns
Customer number	MASNUM	1-5
Customer name	MASNAM	6-27
Street	MASTRT	28-46
City	MASCTY	47-57
State	MASTAT	58-62
Customer balance	MASBAL	63-70
Date of last payment	PAYDAT	71-76
Date of last purchase	PAYPUR	77-82

The program is to process the master customer file, using records from the transaction file, to produce printed receipts. The master file is updated, by producing a new master customer file. The coding required for this program is shown in Figure 101.



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

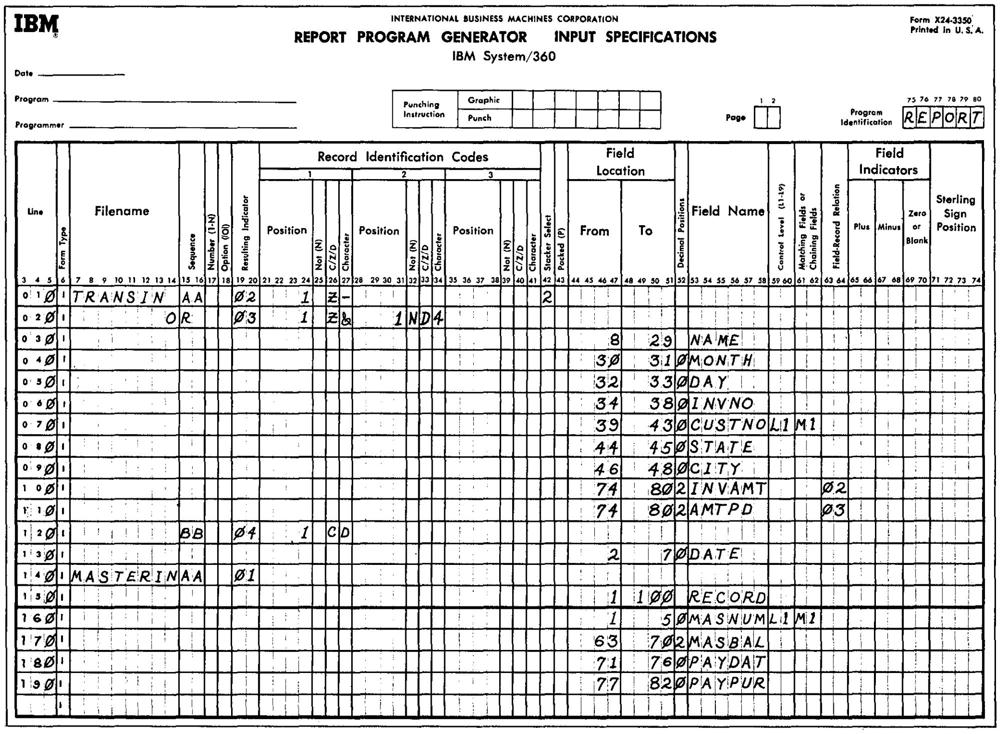
Date
Program
Programmer

Punching Instruction: Graphic / Punch

Page

Program Identification: REPORT

Line	Form Type	Filename	File Type	File Designation	End of File	Sequence	File Format	Block Length	Record Length	Overflow Indicator	Device	Symbolic Device
01Ø	F	TRANSIN	I	S		A	F	8Ø	8Ø		READ4Ø	SYSRDR
02Ø	F	MASTERIN	I	P	E	A	F	3ØØ	1ØØ		TAPE	SYSØØ1
03Ø	F	MASTEROT	O				F	3ØØ	1ØØ		TAPE	SYSØØ2
04Ø	F	MASTLIST	O				V	132	132	OF	PRINTER	SYSLST



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

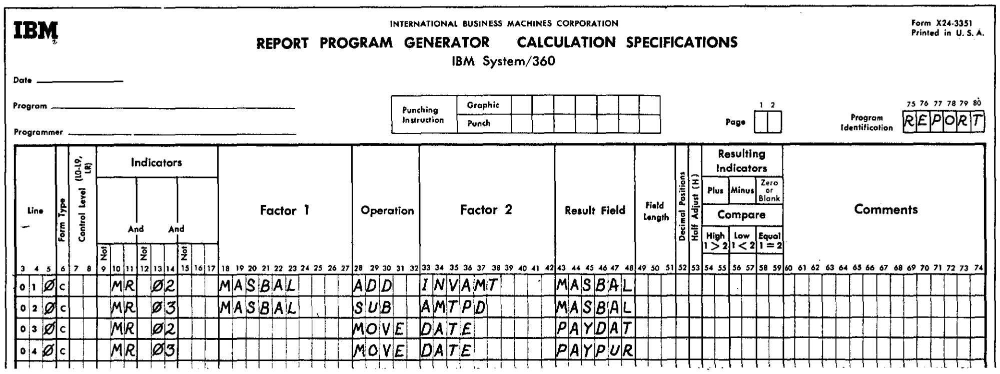
Date
Program
Programmer

Punching Instruction: Graphic / Punch

Page

Program Identification: REPORT

Line	Form Type	Filename	Sequence	Resulting Indicator	Position (1)	C/Z/D (1)	Character (1)	Position (2)	Not (N) (2)	C/Z/D (2)	Character (2)	Stacker Select	From	To	Decimal Positions	Field Name	Control Level	Matching Fields	Field-Record Relation
01Ø	I	TRANSIN	AA	Ø2	1	Z	-					2							
02Ø	I	O	R	Ø3	1	Z	&	1	N	D	4								
03Ø	I												8	29		NAME			
04Ø	I												3Ø	31	Ø	MONTH			
05Ø	I												32	33	Ø	DAY			
06Ø	I												34	38	Ø	INVNO			
07Ø	I												39	43	Ø	CUSTNO	L1	M1	
08Ø	I												44	45	Ø	STATE			
09Ø	I												46	48	Ø	CITY			
10Ø	I												74	8Ø	2	INVAMT			Ø2
11Ø	I												74	8Ø	2	AMTPD			Ø3
12Ø	I		BB	Ø4	1	C	D												
13Ø	I												2	7	Ø	DATE			
14Ø	I	MASTERIN	AA	Ø1															
15Ø	I												1	1ØØ		RECORD			
16Ø	I												1	5	Ø	MASNUM	L1	M1	
17Ø	I												63	7Ø	2	MASBAL			
18Ø	I												71	76	Ø	PAYDAT			
19Ø	I												77	82	Ø	PAYPUR			



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date
Program
Programmer

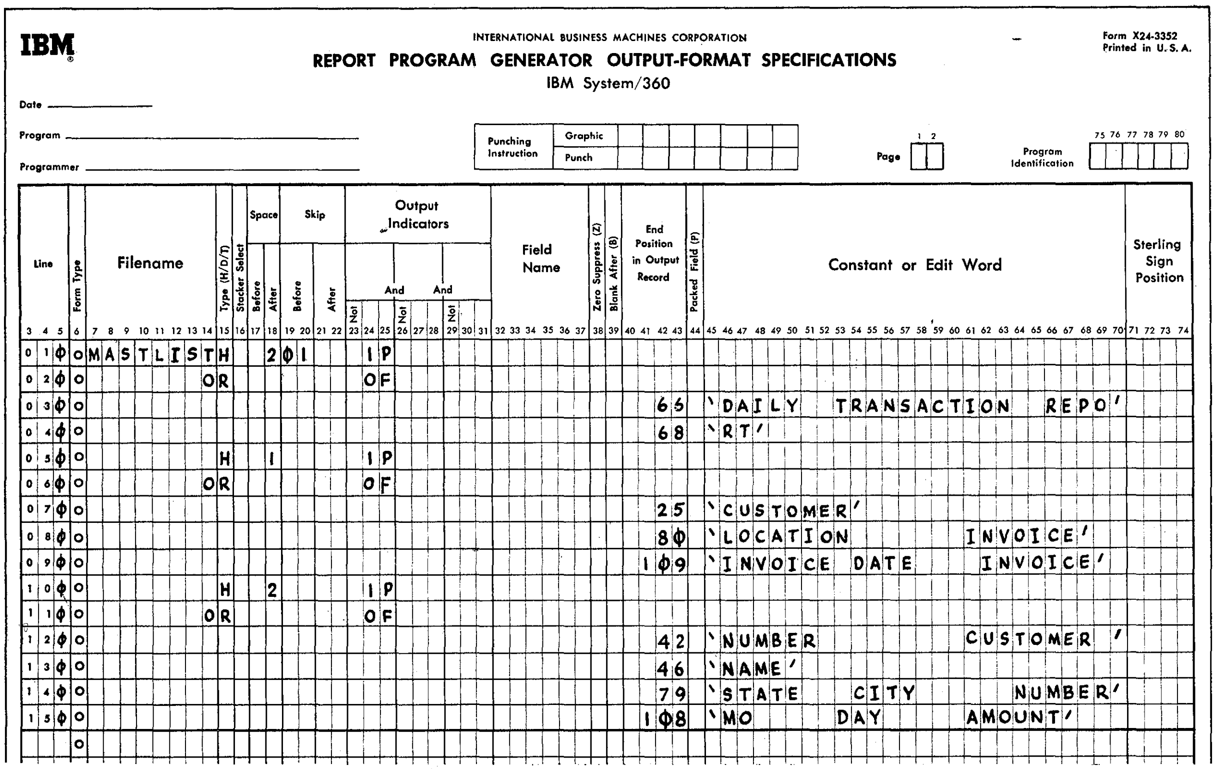
Punching Instruction: Graphic / Punch

Page

Program Identification: REPORT

Line	Form Type	Indicators	And	Factor 1	Operation	Factor 2	Result Field
01Ø	C	MR	Ø2	MASBAL	ADD	INVAMT	MASBAL
02Ø	C	MR	Ø3	MASBAL	SUB	AMTPD	MASBAL
03Ø	C	MR	Ø2		MOVE	DATE	PAYDAT
04Ø	C	MR	Ø3		MOVE	DATE	PAYPUR

Figure 101. Specifications for Sample Program Two (Part 1 of 2)



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

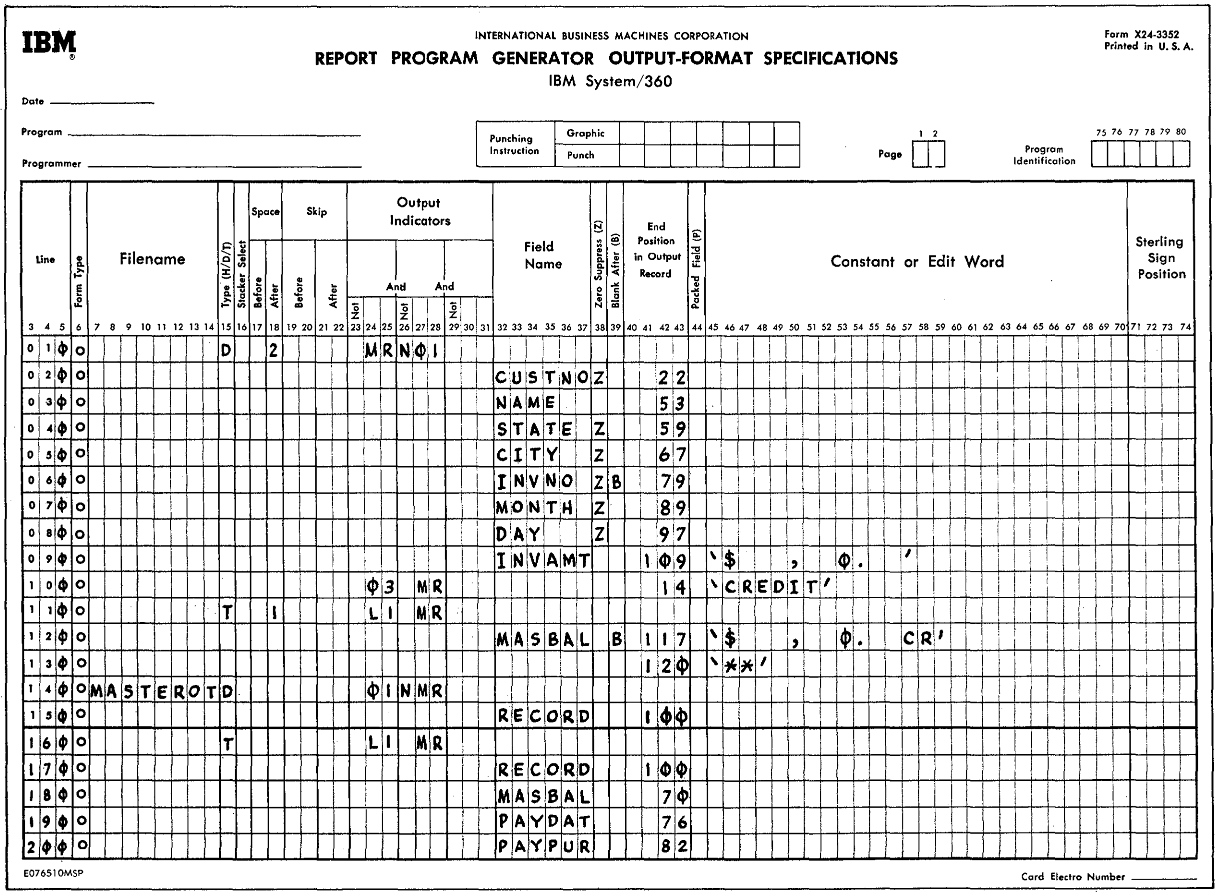
IBM System/360

Form X24-3352
Printed in U.S.A.

Date ____ Program ____ Programmer ____

Punching Instruction: Graphic / Punch — Page — Program Identification

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Not	Output Indicators	Not	And	Not	And	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01Φ	O	MASTLIST	H			2	Φ1			1P											
02Φ	O	OR								OF											
03Φ	O																	65		'DAILY TRANSACTION REPO'	
04Φ	O																	68		'RT'	
05Φ	O		H			1				1P											
06Φ	O	OR								OF											
07Φ	O																	25		'CUSTOMER'	
08Φ	O																	8Φ		'LOCATION INVOICE'	
09Φ	O																	1Φ9		'INVOICE DATE INVOICE'	
1ΦΦ	O		H			2				1P											
11Φ	O	OR								OF											
12Φ	O																	42		'NUMBER CUSTOMER '	
13Φ	O																	46		'NAME'	
14Φ	O																	79		'STATE CITY NUMBER'	
15Φ	O																	1Φ8		'MO DAY AMOUNT'	
	O																				



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U.S.A.

Date ____ Program ____ Programmer ____

Punching Instruction: Graphic / Punch — Page — Program Identification

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Not	Output Indicators	Not	And	Not	And	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01Φ	O		D			2				MR	N	Φ1									
02Φ	O														CUSTNO	Z		22			
03Φ	O														NAME			53			
04Φ	O														STATE	Z		59			
05Φ	O														CITY	Z		67			
06Φ	O														INVNO	Z	B	79			
07Φ	O														MONTH	Z		89			
08Φ	O														DAY	Z		97			
09Φ	O														INVAMT			1Φ9		'$, Φ. '	
1ΦΦ	O									Φ3		MR						14		'CREDIT'	
11Φ	O		T			1				L1		MR									
12Φ	O														MASBAL		B	117		'$, Φ. CR'	
13Φ	O																	12Φ		'**'	
14Φ	O	MASTEROT	D							Φ1	N	MR									
15Φ	O														RECORD			1ΦΦ			
16Φ	O		T							L1		MR									
17Φ	O														RECORD			1ΦΦ			
18Φ	O														MASBAL			7Φ			
19Φ	O														PAYDAT			76			
2ΦΦ	O														PAYPUR			82			

E076510MSP

Card Electro Number ____

Figure 101. Specifications for Sample Program Two (Part 2 of 2)

FILE DESCRIPTION SPECIFICATION SHEET

The four files are defined on this sheet. The two input files TRANSIN and MASTERIN are defined, and the two output files MASTEROT, which is the updated master file, and MASTLIST, which is the printed report, are defined on the file description sheet. TRANSIN is designated as the secondary file because it may not contain transactions involving all the customers on the master customer file. TRANSIN is ascending in order. It has fixed-length records which are 80 characters long. The block length is 80.

MASTERIN is the primary file. If the transaction file does not have a corresponding customer number (specified by the matching fields specification on the input sheet) the master file is processed. Processing continues until all the records in the master customer file have been processed (indicated by the E in column 17). The input records contained in the master customer file are ascending in order, fixed in length, and have a block length of 300. Each record is 100 characters in length.

> Note: A blank in column 53 indicates that there are no labels in the input file.

The file MASTLIST is the printed report. The format is variable. The length of the records can be 132. The overflow entry in columns 33-34 indicate that the overflow condition is to occur on the MASTLIST printer.

The output file MASTEROT is a tape file which contains the updated master customer records. The output records are fixed in length, and are 100 characters in length. The block length is 300.

> Note: A blank in column 53 indicates that there are no labels in the output file.

INPUT SPECIFICATION SHEET

The two input files TRANSIN and MASTERIN are defined on the Input Specification sheet.

TRANSIN: The input records may be obtained from three types of cards. Sequence AA has been assigned to two types. If card column 1 contains the zone of a minus, resulting indicator 02 is turned on. If card column 1 contains the zone of a plus and no digit 4 underpunched, indicator 03 is turned on. The cards that have a minus in column 1 are selected into the 2 pocket (Column 42). The locations of the input records and their labels are defined in columns 46-58 of the sheet.

The field CUSTNO (customer number) has entries in columns 59-60 (Control Level) and columns 61-62 (Matching Fields) of the Input Specification sheet. Whenever a new customer number is read in, control level 1 (L1) is set on. This condition is tested on the output-format specification sheet to govern printing of total lines and to produce the updated customer file. The entry in matching fields specifies that customer number will be used to match another field (MASNUM) in the MASTERIN file.

The fields INVAMT and AMTPD contain entries in column 63-64. Entering 02 in this column indicates that whenever resulting indicator 02 is on, the field INVAMT will be used for input data. Whenever 03 is on, the field AMTPD will be used.

The first card in the transaction file is a data card. It is assigned sequence BB. Whenever column 1 contains a D, indicator 04 is turned on. The date is contained in columns 2-5 of the card.

MASTERIN: The tape input file that contains information about the firm's customers is assigned sequence AA. The first entry under field name defines the entire record. This entry (RECORD) is made to enable the entire record to be referenced on the Output-Format Specification sheet. MASNUM corresponds to CUSTNO in the transaction file. Whenever a new master number is read in, L1 is set. The entry M1 indicates that the master number will be matched with the customer number in the transaction file.

CALCULATION SPECIFICATION SHEET

Whenever the matching record indicator MR is on and indicator 02 is on, the contents of the field INVAMT are added to the MASBAL. The result is stored in MASBAL. The date is moved to the field PAYDAT.

Whenever the matching record indicator MR is on and indicator 03 is on, AMTPD is subtracted from MASBAL, and the result is stored in MASBAL. The date is moved to PAYPUR.

OUTPUT-FORMAT SPECIFICATION SHEET

The output file MASTEROT is the updated tape file. The entries in output indicators allow for the following. Whenever conditions 01 and NMR are satisfied (resulting indicator 01 is on and no matching record is present), the entire tape input record will be written out on tape. This condition results because there was no corresponding customer number in the transaction file for the master customer number.

To keep the master customer file complete, the old input record is written out on the updated tape file when no information is present in the transaction file.

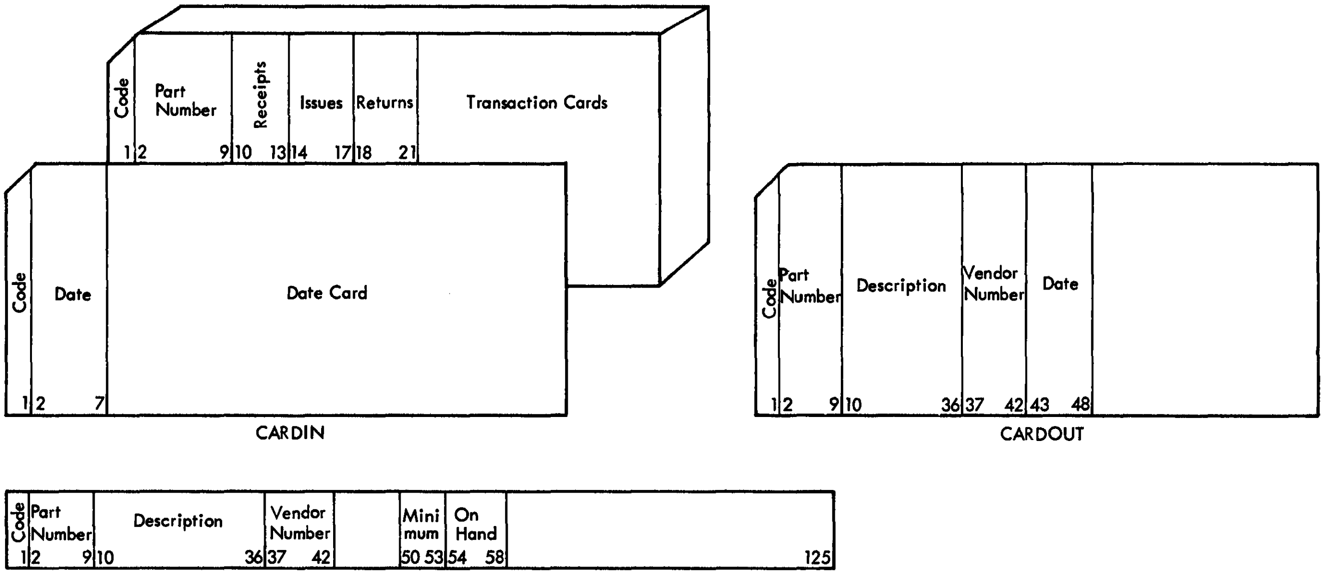
If, however, L1 and MR are on, the input record is written out on tape. 'The entire record is written, but the fields MASBAL, PAYDAT, and PAYPUR contain the new entries based on the calculations. By coding the entries in this way, the new information for MASBAL, PAYDAT, and PAYPUR is entered on the master customer file, but the customer's name and address are retained.

The specifications for the printed report are listed under the name of the output file MASTLIST.

SAMPLE PROGRAM THREE

The third sample program (Figures 102 and 103) illustrates some of the more complex capabilities of PRG.

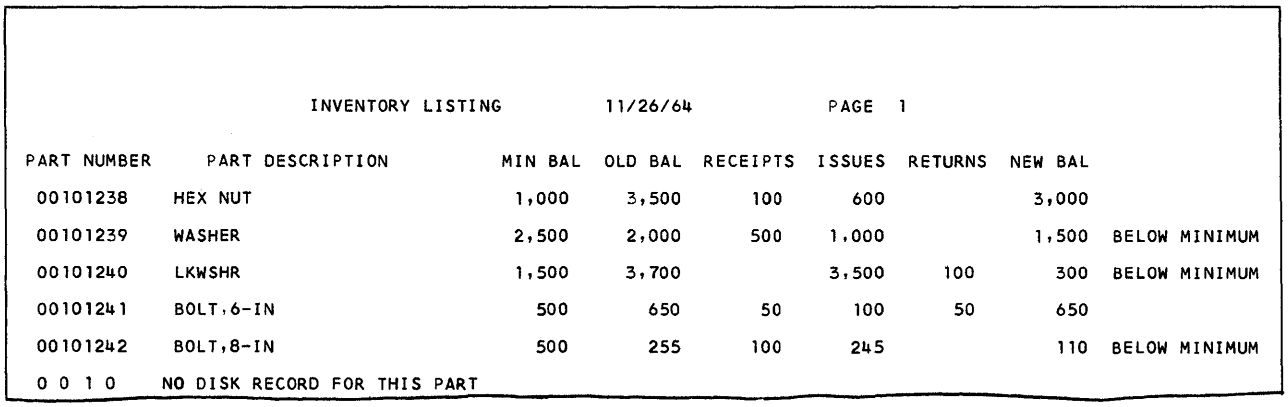
1. Processing chained records on a Direct Access Storage Device (DASD).
2. Updating records on a DASD.
3. Multiple input and output files.
4. Creating exception records
5. Providing for processing when a record in the chained file is missing.



CARDIN

CARDOUT

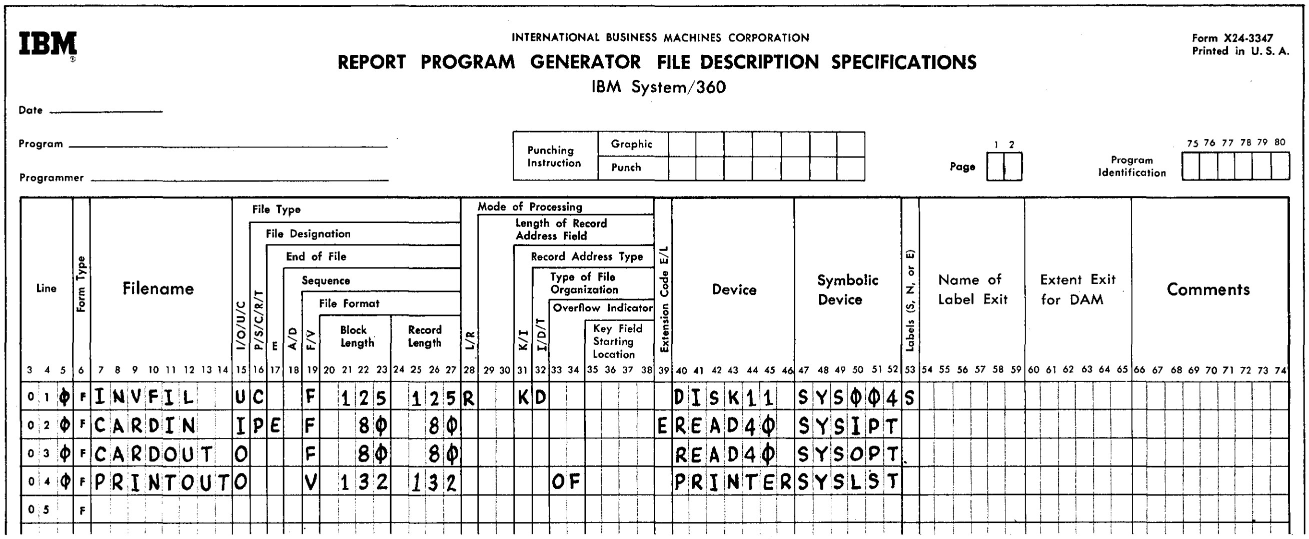
INVFIL



```
                         INVENTORY LISTING        11/26/64          PAGE  1

PART NUMBER      PART DESCRIPTION           MIN BAL   OLD BAL  RECEIPTS  ISSUES  RETURNS  NEW BAL

 00101238    HEX NUT                         1,000     3,500      100      600             3,000
 00101239    WASHER                          2,500     2,000      500    1,000             1,500  BELOW MINIMUM
 00101240    LKWSHR                          1,500     3,700             3,500     100       300  BELOW MINIMUM
 00101241    BOLT,6-IN                         500       650       50      100      50       650
 00101242    BOLT,8-IN                         500       255      100      245               110  BELOW MINIMUM
 0 0 1 0    NO DISK RECORD FOR THIS PART
```

PRINTOUT

Figure 102. Input and Output Formats for Sample Program Three



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

Form X24-3347
Printed in U. S. A.

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Date ____________

Program ____________

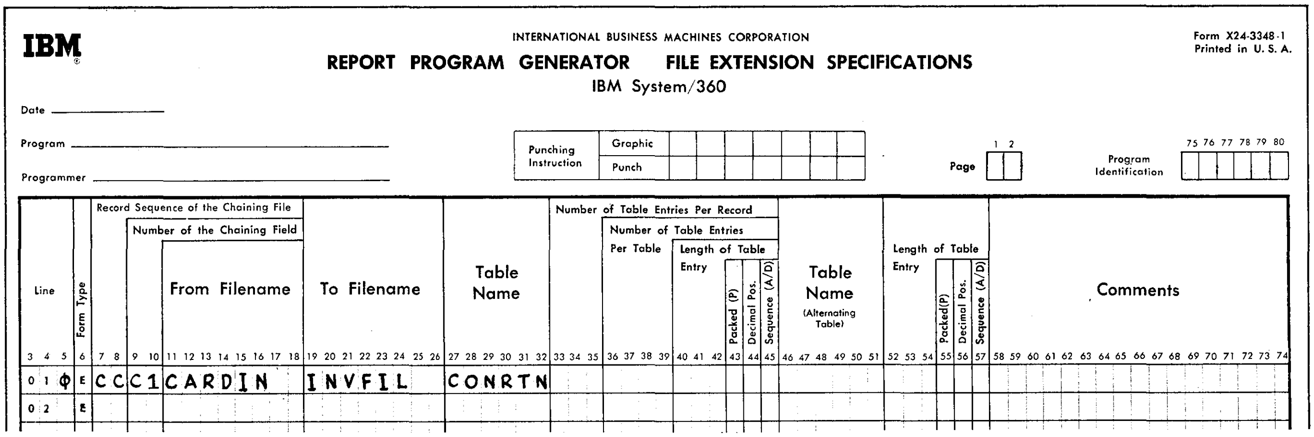
Programmer ____________

Punching Instruction	Graphic							
	Punch							

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3 4 5)	Form Type (6)	Filename (7–14)	File Type: I/O/U/C (15)	File Designation: P/S/C/R/T (16)	End of File: E (17)	Sequence: A/D (18)	File Format: F/V (19)	Block Length (20–23)	Record Length (24–27)	Mode of Processing: L/R (28)	Length of Record Address Field (29–30)	Record Address Type: K/I (31)	Type of File Organization: I/D/T (32)	Overflow Indicator (33–34)	Key Field Starting Location (35–38)	Extension Code E/L (39)	Device (40–46)	Symbolic Device (47–52)	Labels (S, N, or E) (53)	Name of Label Exit (54–59)	Extent Exit for DAM (60–65)	Comments (66–74)
01Φ	F	INVFIL	U	C			F	125	125	R		K	D				DISK11	SYSΦΦ4	S			
02Φ	F	CARDIN	I	P	E		F	8Φ	8Φ							E	READ4Φ	SYSIPT				
03Φ	F	CARDOUT	O				F	8Φ	8Φ								READ4Φ	SYSOPT				
04Φ	F	PRINTOUT	O				V	132	132					OF			PRINTER	SYSLST				
05	F																					



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

Form X24-3348-1
Printed in U. S. A.

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Date ____________

Program ____________

Programmer ____________

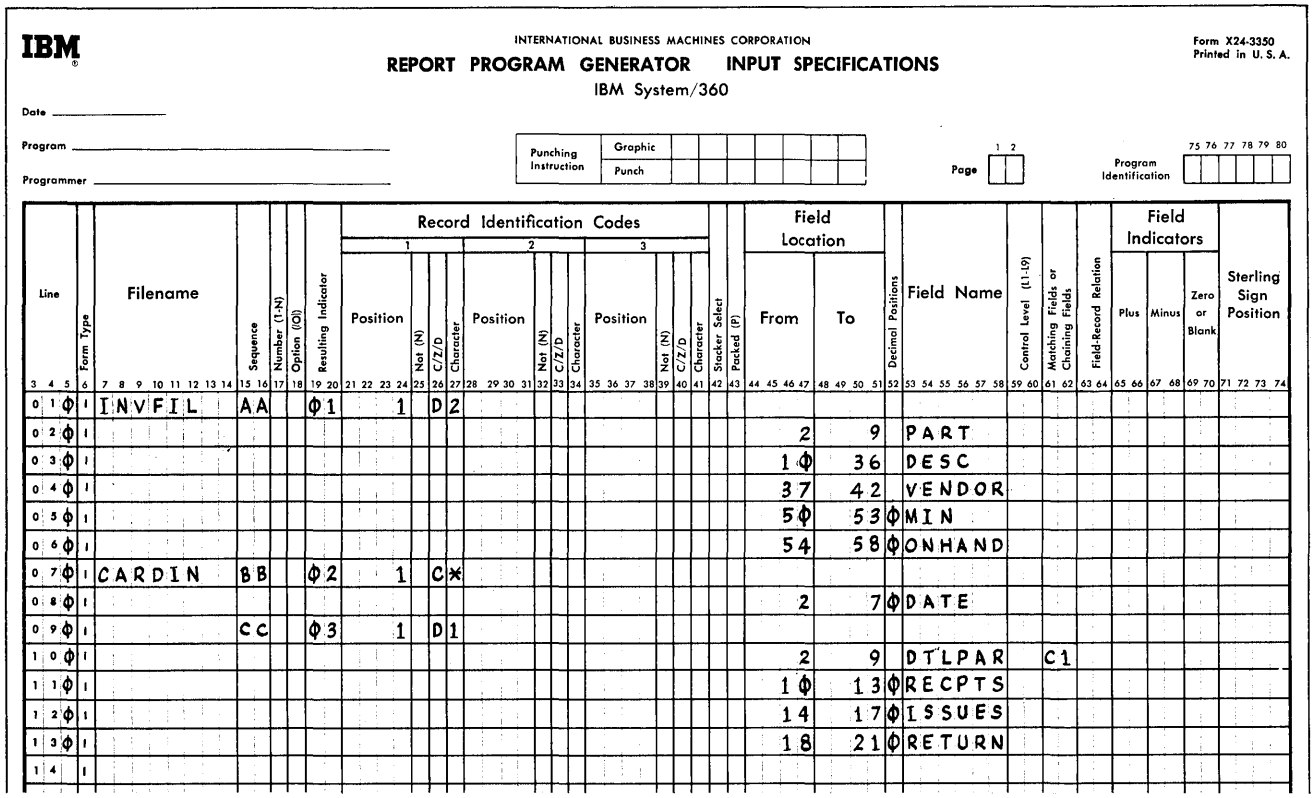
Punching Instruction	Graphic							
	Punch							

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3 4 5)	Form Type (6)	Record Sequence of the Chaining File (7–8)	Number of the Chaining Field (9–10)	From Filename (11–18)	To Filename (19–26)	Table Name (27–32)	Number of Table Entries Per Record (33–35)	Number of Table Entries Per Table (36–39)	Length of Table Entry (40–42)	Packed (P) (43)	Decimal Pos. (44)	Sequence (A/D) (45)	Table Name (Alternating Table) (46–51)	Length of Table Entry (52–54)	Packed(P) (55)	Decimal Pos. (56)	Sequence (A/D) (57)	Comments (58–74)
01Φ	E	CC	C1	CARDIN	INVFIL	CONRTN												
02	E																	

Figure 103. Specifications for Sample Program Three (Part 1 of 4)



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date ______

Program ______

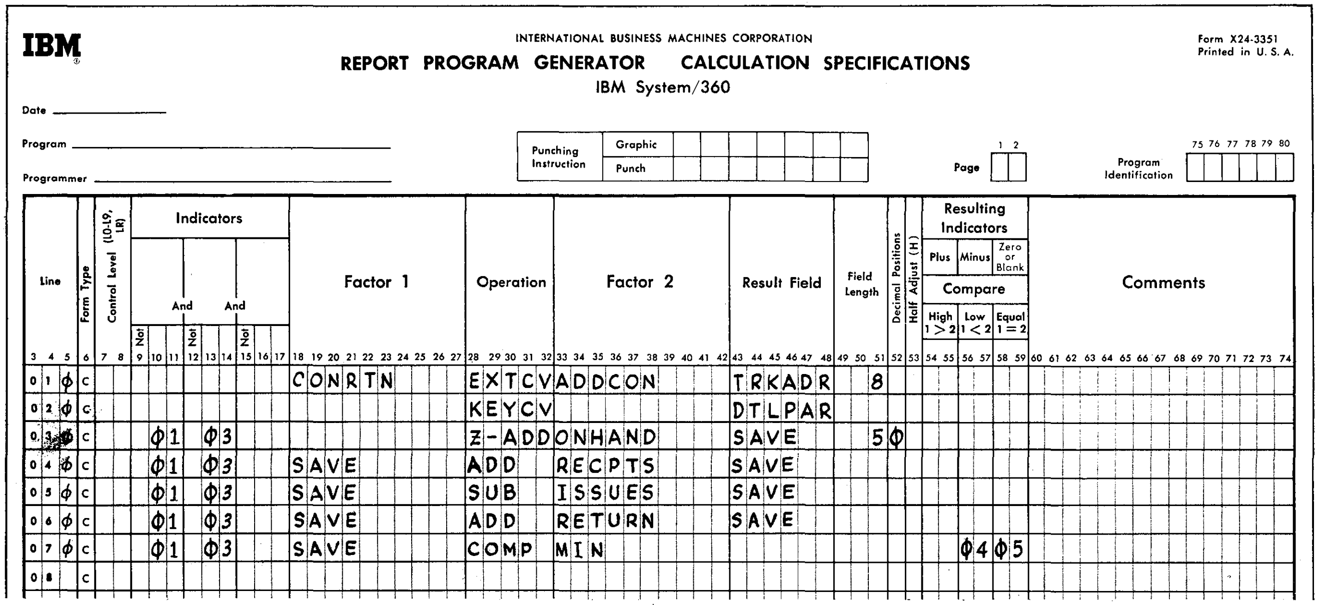
Programmer ______

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Resulting Indicator (19-20)	Record Identification Codes 1: Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	2: Position (28-31)	Not (N) (32)	C/Z/D (33)	Character (34)	3: Position (35-38)	Not (N) (39)	C/Z/D (40)	Character (41)	Stacker Select (42)	Packed (P) (43)	Field Location From (44-47)	Field Location To (48-51)	Decimal Positions (52)	Field Name (53-58)	Control Level (L1-L9) (59-60)	Matching Fields or Chaining Fields (61-62)	Field-Record Relation (63-64)	Field Indicators Plus (65-66)	Minus (67-68)	Zero or Blank (69-70)	Sterling Sign Position (71-74)
01Φ	I	INVFIL	AA			Φ1	1		D	2																					
02Φ	I																				2	9		PART							
03Φ	I																				1Φ	36		DESC							
04Φ	I																				37	42		VENDOR							
05Φ	I																				5Φ	53	Φ	MIN							
06Φ	I																				54	58	Φ	ONHAND							
07Φ	I	CARDIN	BB			Φ2	1		C	*																					
08Φ	I																				2	7	Φ	DATE							
09Φ	I		CC			Φ3	1		D	1																					
10Φ	I																				2	9		DTLPAR		C1					
11Φ	I																				1Φ	13	Φ	RECPTS							
12Φ	I																				14	17	Φ	ISSUES							
13Φ	I																				18	21	Φ	RETURN							
14	I																														



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Form X24-3351
Printed in U. S. A.

Date ______

Program ______

Programmer ______

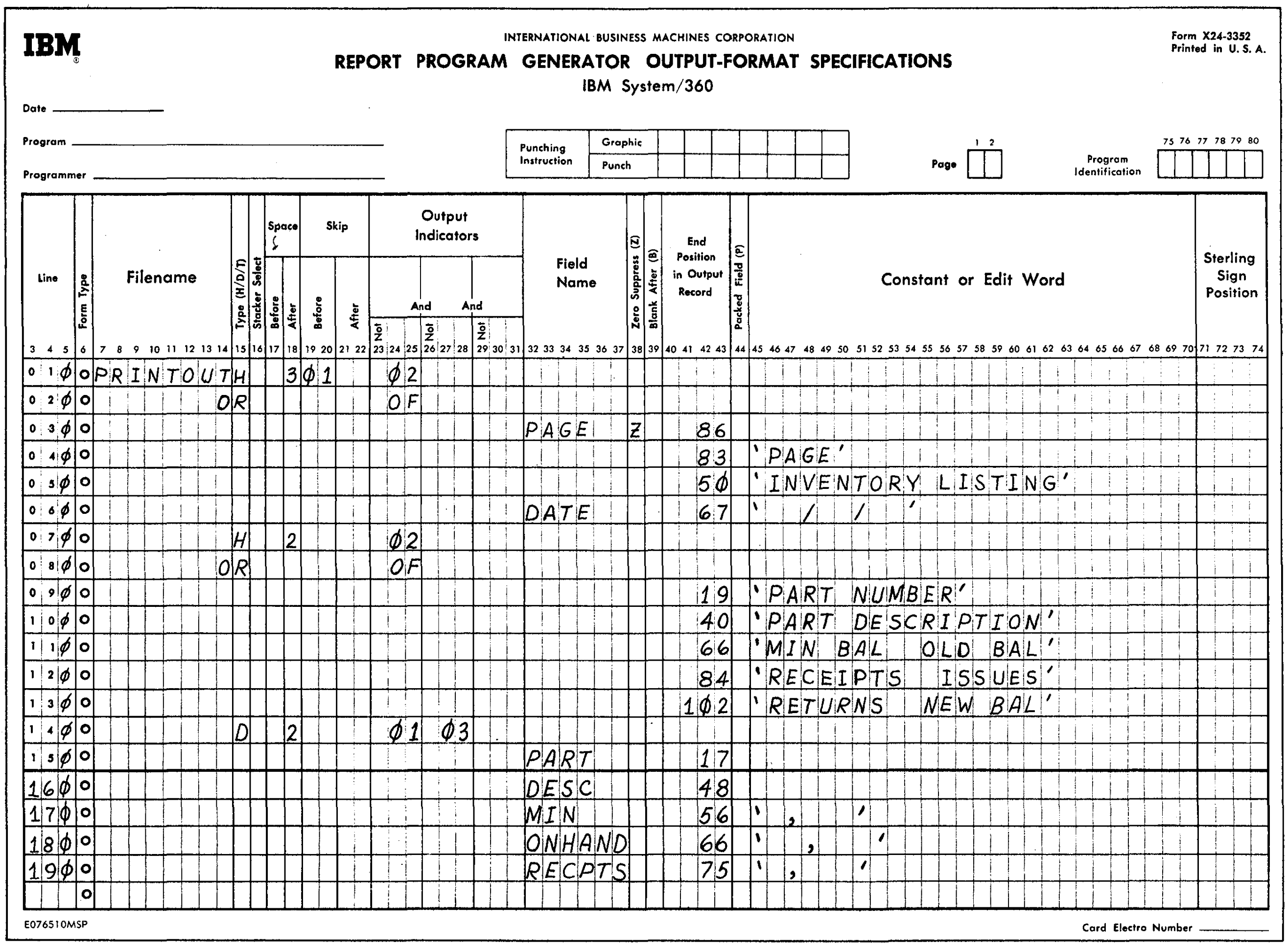
Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR) (7-8)	Indicators Not (9)	(10-11)	And Not (12)	(13-14)	And Not (15)	(16-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators Plus / Compare High 1 > 2 (54-55)	Minus / Low 1 < 2 (56-57)	Zero or Blank / Equal 1 = 2 (58-59)	Comments (60-74)
01ϕ	C								CONRTN	EXTCV	ADDCON	TRKADR	8						
02ϕ	C									KEYCV		DTLPAR							
03ϕ	C			Φ1		Φ3				Z-ADD	ONHAND	SAVE	5	Φ					
04ϕ	C			Φ1		Φ3			SAVE	ADD	RECPTS	SAVE							
05ϕ	C			Φ1		Φ3			SAVE	SUB	ISSUES	SAVE							
06ϕ	C			Φ1		Φ3			SAVE	ADD	RETURN	SAVE							
07ϕ	C			Φ1		Φ3			SAVE	COMP	MIN						Φ4	Φ5	
08	C																		

● Figure 103. Specifications for Sample Program Three (Part 2 of 4)



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

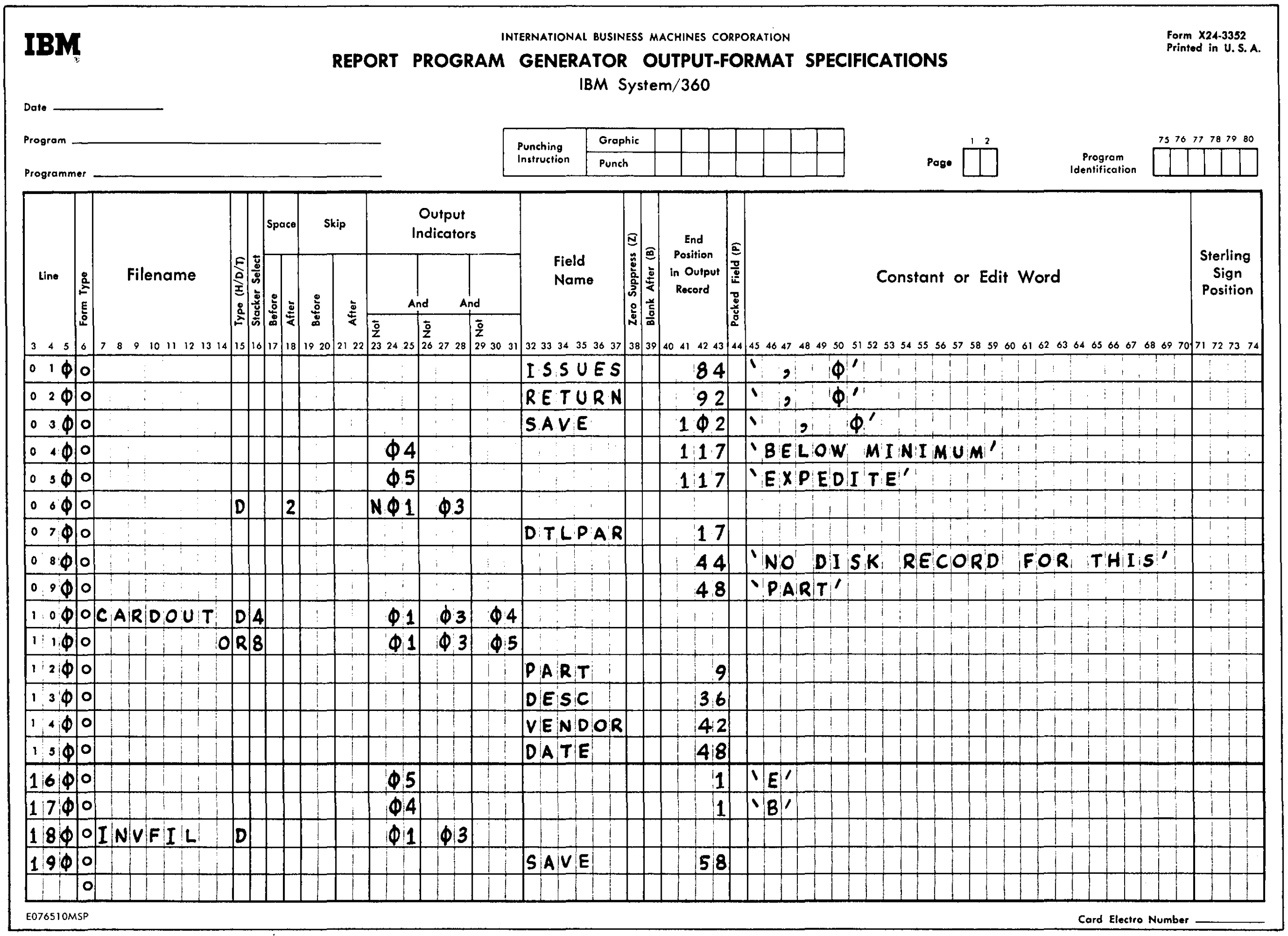
Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators Not (23)	(24-25)	And Not (26)	(27-28)	And Not (29)	(30-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)	End Position in Output Record (40-43)	Packed Field (P) (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
01Ø	O	PRINTOUT	H			3	Ø1			Ø2											
02Ø	O		OR							OF											
03Ø	O														PAGE	Z		86			
04Ø	O																	83		'PAGE'	
05Ø	O																	5Ø		'INVENTORY LISTING'	
06Ø	O														DATE			67		' / / '	
07Ø	O		H			2				Ø2											
08Ø	O		OR							OF											
09Ø	O																	19		'PART NUMBER'	
10Ø	O																	40		'PART DESCRIPTION'	
11Ø	O																	66		'MIN BAL OLD BAL'	
12Ø	O																	84		'RECEIPTS ISSUES'	
13Ø	O																	1Ø2		'RETURNS NEW BAL'	
14Ø	O		D			2				Ø1		Ø3									
15Ø	O														PART			17			
16Ø	O														DESC			48			
17Ø	O														MIN			56		' , '	
18Ø	O														ONHAND			66		' , '	
19Ø	O														RECPTS			75		' , '	
	O																				

E076510MSP

Card Electro Number

● Figure 103. Specifications for Sample Program Three (Part 3 of 4)



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24-3352
Printed in U. S. A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T) (15)	Stacker Select (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators Not (23-25)	And Not (26-28)	And Not (29-31)	Field Name (32-37)	Zero Suppress (Z) (38)	Blank After (B) (39)	End Position in Output Record (40-43)	Packed Field (P) (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
01Φ	O											ISSUES			84		' , Φ'	
02Φ	O											RETURN			92		' , Φ'	
03Φ	O											SAVE			1Φ2		' , Φ'	
04Φ	O								Φ4						117		'BELOW MINIMUM'	
05Φ	O								Φ5						117		'EXPEDITE'	
06Φ	O		D			2			NΦ1	Φ3								
07Φ	O											DTLPAR			17			
08Φ	O														44		'NO DISK RECORD FOR THIS'	
09Φ	O														48		'PART'	
10Φ	O	CARDOUT	D	4					Φ1	Φ3	Φ4							
11Φ	O	O	R	8					Φ1	Φ3	Φ5							
12Φ	O											PART			9			
13Φ	O											DESC			36			
14Φ	O											VENDOR			42			
15Φ	O											DATE			48			
16Φ	O								Φ5						1		'E'	
17Φ	O								Φ4						1		'B'	
18Φ	O	INVFIL	D						Φ1	Φ3								
19Φ	O											SAVE			58			
	O																	

E076510MSP

Card Electro Number

Figure 103. Specifications for Sample Program Three (Part 4 of 4)

FILE DESCRIPTION SPECIFICATIONS

The primary input file, designated by a P in column 16, is on the card reader and is labeled CARDIN. This card file also acts as a transaction file for the updating of the inventory records contained on the file labeled INVFIL. The U in column 15 of the specification for INVFIL indicates that this DASD file will be updated (used for both input and output). The file is organized under direct access management, indicated by the D in column 32, and will be processed randomly as specified in column 28.

The E in column 17 of the CARDIN file description entry directs the program to end execution when the last record of the CARDIN file has been processed. The E in column 39 indicates that a file-extension specification has been coded for the CARDIN file.

The output files are an exception card file labeled CARDOUT and the printed transaction report labeled PRINTOUT.

CARDIN and CARDOUT consist of 80-character records. INVFIL has a block length of 125 characters and a record length of 125 characters. Record length for PRINTOUT is variable, with a maximum of 132 characters.

FILE EXTENSION SPECIFICATIONS

The file-extension sheet identifies CARDIN as the chaining file, indicating that record sequence CC contains the chaining field C1. INVFIL is the chained file and CONRTN is the RPG internal label of the specifications on the Calculation Specification sheet for the external conversion routine that processes the chaining field to obtain the track address in the DASD file.

INPUT SPECIFICATIONS

The input card and DASD files are described on the Input Specification sheet. Each field that will be used is defined. The C1 in columns 61-62 of the transaction card file designates the field labeled DTLPAR as the chaining field. The conversion routine CONRTN, indicated on the Calculation Specification sheet, will make the DTLPAR field available to the user's external conversion routine.

CALCULATION SPECIFICATIONS

Two sets of calculation specifications are described. After the label CONRTN, the operation code EXTCV is used to indicate that the conversion routine is to be performed. ADDCON is the symbolic address of the external conversion routine. Note that the address can be up to eight positions long. The external routine may be in the relocatable library so that it can be automatically retrieved by the Basic Operating System program. The routine could also be punched in cards that follow the RPG cards.

The operation code KEYCV states that the record key can be obtained directly from the field DTLPAR. This is the first entry on the Calculation Specification sheet.

The calculations in lines 030-070 require the presence of matching chaining (CARDIN) and chained (INVFIL) records. This condition is indicated by the simultaneous setting of resulting indicators 01 and 03. The ONHAND field from the INVFIL record is reset added to a work area (SAVE). Data fields RECPTS and RETURN from CARDIN records are added to SAVE; ISSUES are subtracted. When each input card for an INVFIL record is processed, the resulting new ONHAND field in SAVE is compared to the minimum balance. If ONHAND equals the minimum, resulting indicator 05 is set on. If ONHAND drops below the minimum, resulting indicator 04 is set on.

OUTPUT SPECIFICATIONS

The PRINTOUT file's heading information can be printed under control of either resulting indicator 02, which is set for the date card (the first card in the CARDIN file), or overflow (OF). The OF indicator will govern all heading printing after the first page. The entry PAGE in line 030 causes the page number to be updated automatically for each new page.

The detail line described by the entries in lines 150 of Figure 103 (Part 3) through 050 of Figure 103 (Part 4) requires the presence of both the CARDIN and INVFIL records (resulting indicators 01 and 03 on). If resulting indicator 04 is on, the words BELOW MINIMUM indicate the stock violation. If indicator 05 is on, the message EXPEDITE is added.

The output line described in PRINTOUT entries 060-090 is the message printed when the error condition of a CARDIN record, and no corresponding INVFIL record occurs. This condition is identified by resulting indicator 01 being off when indicator 03 is on.

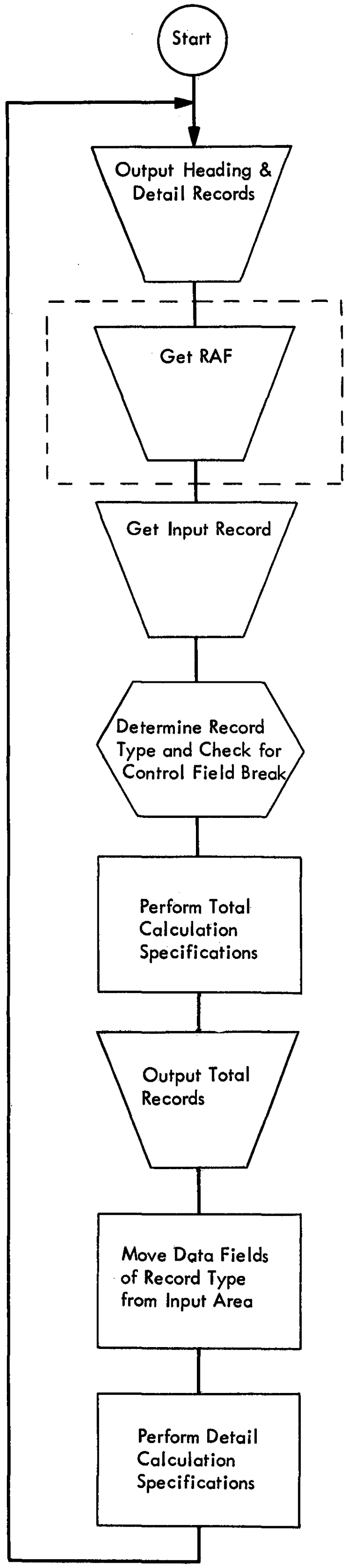
The exception file CARDOUT will have a card punched for each transaction that results in a below-minimum or at-minimum stock level. These cards will contain the part number and description, vendor number, date, and the E or B code for EXPEDITE or BELOW MINIMUM. Below-minimum cards, identified by the simultaneous ON settings of indicators 01, 03, and 04, will be selected to stacker 4. At-minimum cards, with indicators 01, 03, and 05 on, will be selected into stacker 8.

Lines 180 and 190 provide for the updating and writing out of the INVFIL records. If indicators 01 and 03 are both on, indicating that the INVFIL record has been updated by a CARDIN transaction, the new ONHAND is moved into its INVFIL location from the work area SAVE, and the record is written.

Indicators	Where Specified	Where Used	Turned On	Turned Off	Notes
Resulting Indicator 00	Internal	Output Indicators on Output Specifications (Columns 24 - 25, 27 - 28, 30 - 31)	This indicator is always ON	Can never be turned OFF	
Resulting Indicators (01-99)	Input Specifications (Columns 19-20)	Indicators on Calculation Specifications (Columns 10-11, 13-14, 16-17) Output Indicators on Output Specifications (Columns 24-25, 27-28, 30-31)	When specified record type has been read and is ready for processing.	Before the first record is read on the next processing cycle.	Turning OFF and ON can also be accomplished by using SETON and SETOF operation codes.
	Calculation Specifications (Columns 54-55, 56-57, 58-59)	Same as above.	Whenever the specified field status condition is satisfied.	The next time that this field status is to be tested	Same as above
Field Indicators (01-99)	Input Specifications (Columns 65-66, 67-68, 69-70)	Same as above	Same as above	Same as above	Same as above
Halt Indicators (H1 - H9)	Input Specifications (Columns 65-66, 67-68, 69-70) Calculation Specifications (Columns 54-55, 56-57, 58-59)	Same as above	Same as above	Can only be turned OFF by SETOF operation code. See Note.	If these indicators remain ON, the object program will terminate before reading the next record.
Control Level Indicators (L1 - L9)	Input Specifications (Columns 59-60)	Same as above and Calculation Specifications (Columns 7-8)	When the value in a control field changes. All indicators of the lower levels are also turned ON.	Before the first record is read on the next processing cycle.	Turning OFF and ON can be accomplished by using SETON and SETOF operation codes.
L0	Internal	Indicators on Calculations Specifications (Columns 10-11, 13-14, 16-17) Output Indicators on Output Specifications (Cols. 24-25,27-28,30-31)	This indicator is always ON.	Can never be turned OFF.	
LR	Internal	Same as above	After processing the last record of the last file.	Same as above	All Control Level Indicators (L1-L9) are also turned ON when the LR is turned ON.
MR	Internal	Same as above	When multiple input files and the matching fields specification are used, this indicator is turned ON if a secondary file record matches the primary file record.	After all total records have been put out for the last secondary record of a matching group.	
Overflow Indicators OA, OF, OV	File Description (Columns 33-34)	Same as above	When Channel 12 of the carriage control tape is sensed.	After the detail and heading records are written.	These indicators remain ON for one complete processing cycle.
1P	Internal	Output Indicators on Output Specifications (Columns 24-25, 27-28, 30-31)	This indicator is ON at the beginning of processing before any records are read.	Before the first record is read.	This indicator is used to govern printing of the first page of the report.

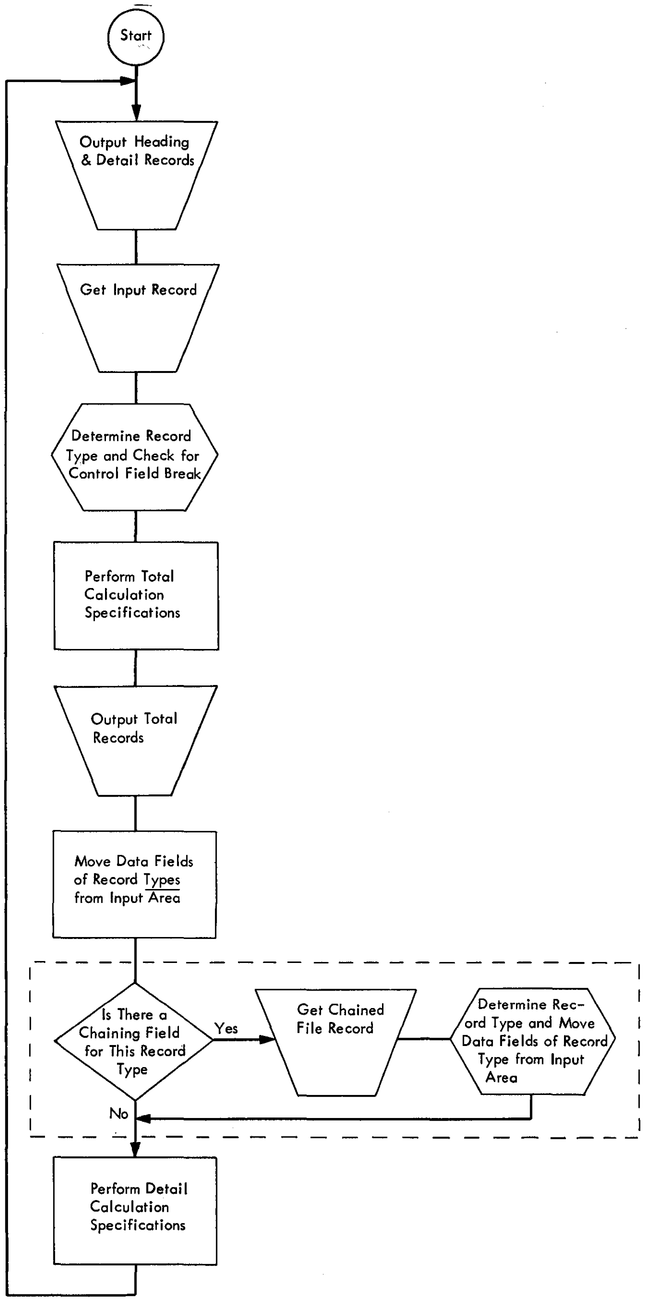
All indicators specified in columns 69 - 70 of the Input Specification sheet and columns 58 - 59 of the Calculation Specification sheet are ON at the beginning of processing before any records are read. This is always the case except when the operation codes COMP, LOKUP, SETON, SETOF and TESTZ are specified.

APPENDIX B: RPG LOGIC FLOW CHARTS



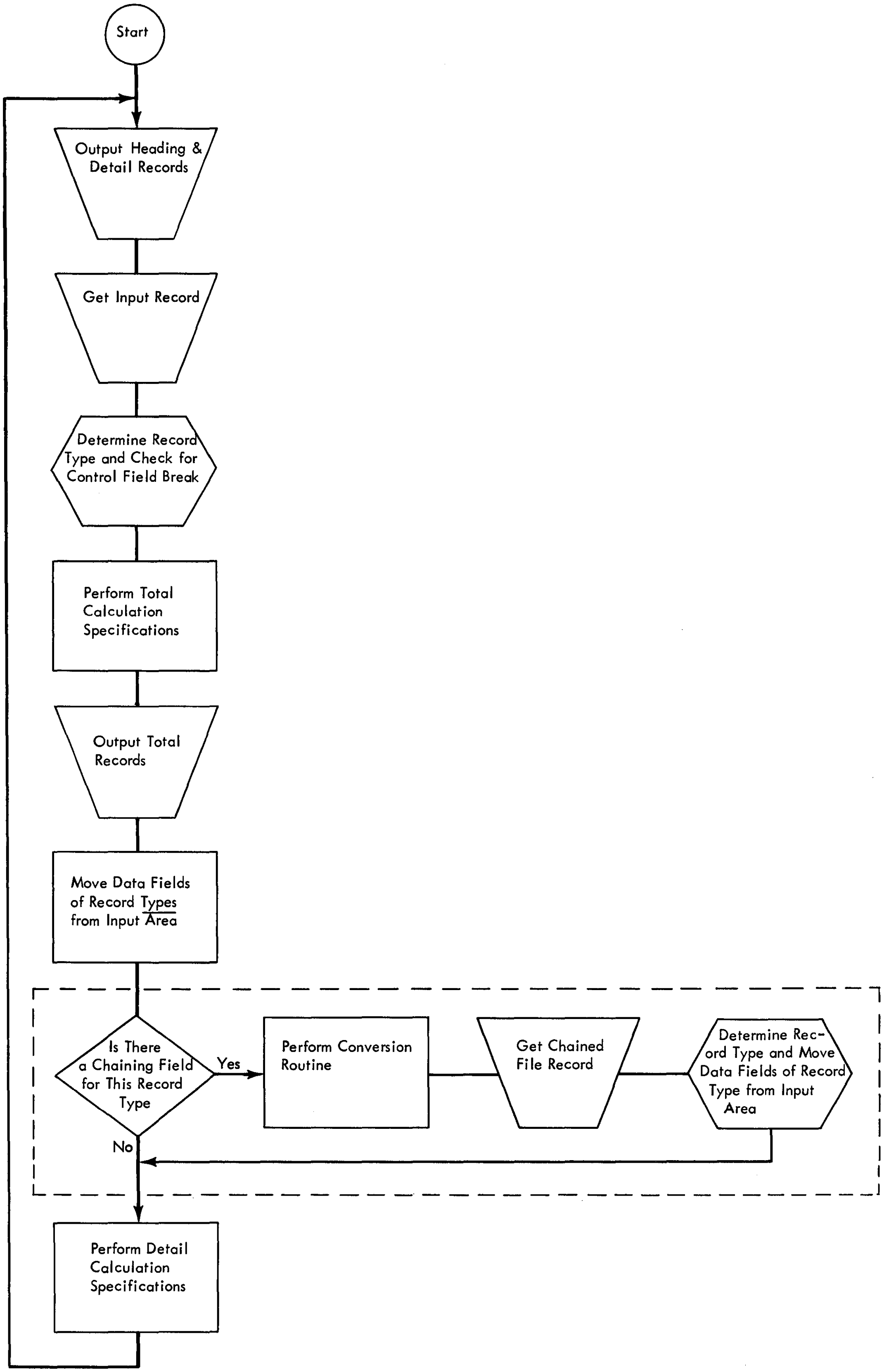
Start
Output Heading & Detail Records
Get RAF
Get Input Record
Determine Record Type and Check for Control Field Break
Perform Total Calculation Specifications
Output Total Records
Move Data Fields of Record Type from Input Area
Perform Detail Calculation Specifications

1. General Logic for RPG Object Program with RAF File

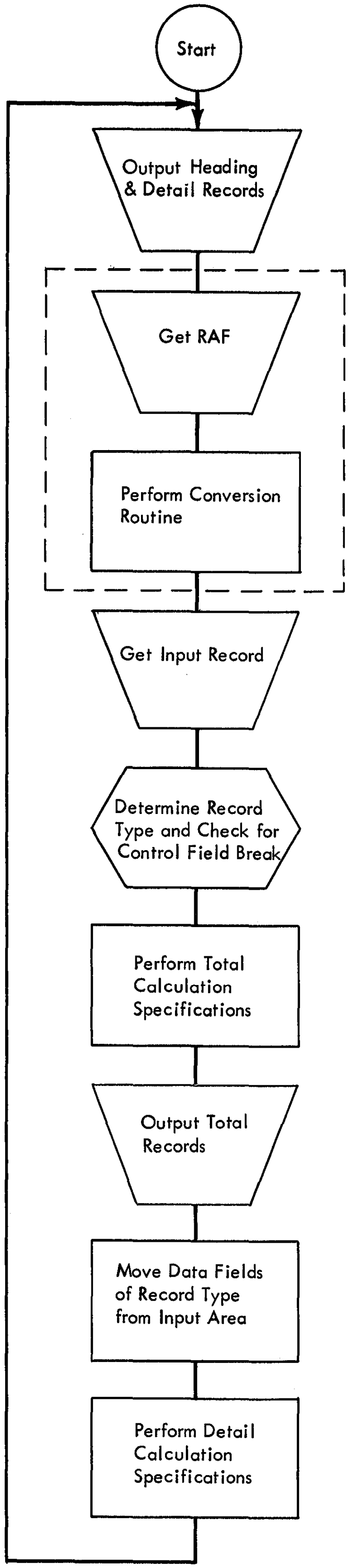


Start
Output Heading & Detail Records
Get Input Record
Determine Record Type and Check for Control Field Break
Perform Total Calculation Specifications
Output Total Records
Move Data Fields of Record Types from Input Area
Is There a Chaining Field for This Record Type
Yes
Get Chained File Record
Determine Record Type and Move Data Fields of Record Type from Input Area
No
Perform Detail Calculation Specifications

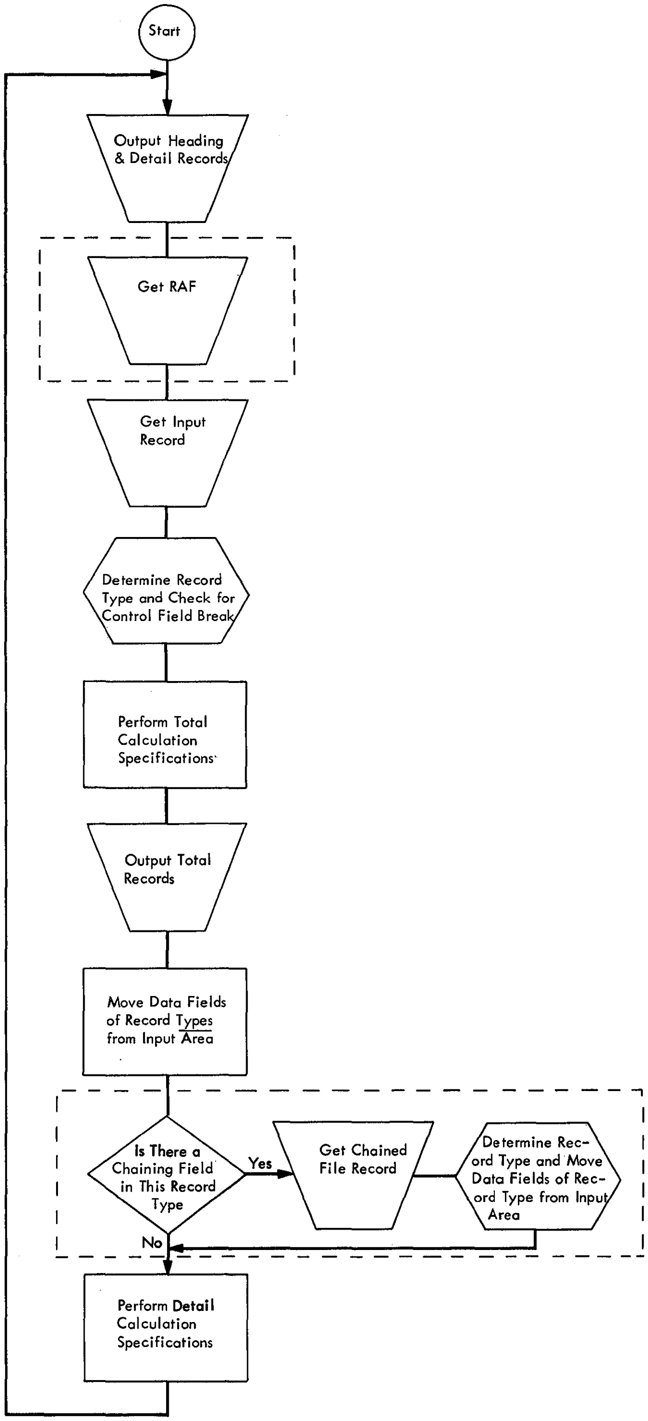
2. General Logic for RPG Object Program with a Chaining File


Start
Output Heading & Detail Records
Get Input Record
Determine Record Type and Check for Control Field Break
Perform Total Calculation Specifications
Output Total Records
Move Data Fields of Record Types from Input Area
Is There a Chaining Field for This Record Type
Yes
No
Perform Conversion Routine
Get Chained File Record
Determine Record Type and Move Data Fields of Record Type from Input Area
Perform Detail Calculation Specifications

3. General Logic for RPG Object Program with Chaining File Which Requires Conversion


Start
Output Heading & Detail Records
Get RAF
Perform Conversion Routine
Get Input Record
Determine Record Type and Check for Control Field Break
Perform Total Calculation Specifications
Output Total Records
Move Data Fields of Record Type from Input Area
Perform Detail Calculation Specifications

4. General Logic for RPG Object Program with RAF File which Requires Conversion


Start
Output Heading & Detail Records
Get RAF
Get Input Record
Determine Record Type and Check for Control Field Break
Perform Total Calculation Specifications
Output Total Records
Move Data Fields of Record Types from Input Area
Is There a Chaining Field in This Record Type
Yes
No
Get Chained File Record
Determine Record Type and Move Data Fields of Record Type from Input Area
Perform Detail Calculation Specifications

5. General Logic for RPG Object Program with RAF and Chaining Files

The RPG sterling routines furnish users with a convenient and time-saving means of handling sterling amounts. The presence of sterling fields is indicated to the RPG program by additional entries in the input and output specifications forms and in the control card. The file description, file extension, and calculation forms are not affected.

Sterling input information can be represented in two formats: IBM and BSI as described in the control card. The RPG sterling routines convert the input fields into a pence-format field. A pence-format field is a sterling amount represented in pence. If the output is to be printed, the fields are converted with shillings and pence printed in two positions each with zero suppression in effect in the tens position of each field. If the output is not printed, the output is converted to either BSI or IBM formats.

> Note BSI or IBM input files of one program must use the same code combination throughout.

INPUT SPECIFICATIONS

The position of the sign must be specified in columns 71-74. Enter an S in column 74, if the sign is in the normal position. If the pence field has decimal positions, the normal position of the sign is in the rightmost decimal position of the pence field. If the pence field has no decimal positions, the normal position of the sign is in the units position of the pounds field.

> Note 1: One of the digits 0, 1, 2, or 3 must be entered in column 52, to indicate the number of required decimal positions.
>
> Note 2: It is not permissible to use the same name for both a sterling field and a decimal field.
>
> Note 3: The sign of the field must contain a numeric underpunch.

OUTPUT WHICH IS NOT PRINTED: Sterling output fields are specified in the same manner as sterling input fields.

PRINTED OUTPUT: Insert the letter S in column 74 of the Output-Format Specification sheet.

> Note 1: Shillings and pence are printed in two positions each, with zero suppression in effect in the tens positions of each field.
>
> Note 2: If a field is defined as a sterling field in the input but not in the output specifications, the output will be in pence format.
>
> Note 3: The rules governing the use of edit control words are the same as those for decimal fields.

The following features are available:

1. Zero suppression in the pounds field.
2. Zero suppression in the shillings field, if both pound and shilling values are zero.
3. Zero suppression in the pence field, if pound, shilling, and penny values are zero.
4. Suppression of zeros preceding signs, and suppression of separation marks between pounds and shillings, shillings and pence, and pence and decimals.

> Note: If column 38 of the Output-Format Specification sheet contains Z, the entire pounds field will be zero-suppressed. The ten's position of the shillings and pence fields will be zero-suppressed, and the sign of the field will be blanked out.

CONTROL CARD

To select the required sterling routines, the RPG program needs information regarding the input and output formats. This information is entered in four columns of the RPG processor control card. The entries are: 1 for IBM Code, 2 for BSI Code.

CALCULATION SPECIFICATIONS

While no additional entries are required in this form, the user should keep in mind that all calculations are done in pence format. This must be considered when defining the length of result fields or when using Factors 1 and 2.

Lengths of Pence-Format Fields

If a pence-format result field is to be reconverted into a Format-1 field, the highest amount it is permitted to contain is 2,399,999,999,999,999.

POUND STERLING FORMATS

RPG will support, on the input and output fields, two standards for pence and shilling portions of sterling fields: IBM or BSI. Columns 17-20 of the RPG Processor Control Card indicate either the IBM or BSI formats. The formats for IBM and BSI are listed here.

COLUMN 17 (STERLING-SHILLING FIELD ON INPUT) IBM FORMAT: Two positions are allowed for the shilling option in the input field: 00-19 for 0 to 19 shillings.

BSI FORMAT: The shilling option in the input fields is indicated as listed here:

1-9 Shillings by a 1-9 punch.

10 Shillings by a 12-punch.

11-19 Shillings by a A-I punch.

COLUMN 18 (STERLING-PENCE FIELD ON INPUT) IBM FORMAT: The pence option on the input field is as listed here:
1-9 Pence by a 1-9 punch,
10 Pence by an 11-punch,
11 Pence by a 12-punch.

BSI FORMAT: The pence option on the input field is as listed here:
1-9 Pence by a 1-9 punch,
10 Pence by a 12-punch,
11 Pence by an 11-punch.

COLUMN 19 (STERLING-SHILLINGS FIELD ON OUTPUT) IBM FORMAT: Two positions are allowed for the shilling option on the output field:
00-19 for 0-19 shillings.

BSI FORMAT: The shilling option on the output field is as listed here:
1-9 Shillings by a 1-9 punch,
10 Shillings by a 12-punch,
11-19 Shillings by A-I punch,

COLUMN 20 (STERLING-PENCE FIELD ON OUTPUT) IBM FORMAT: The pence option on the output field is as listed here:
1-9 Pence by a 1-9 punch,
10 Pence by an 11-punch,
11 Pence by a 12-punch.

BSI FORMAT: The pence option on the output field is as listed here:
1-9 Pence by a 1-9 punch.
10 Pence by a 12 punch,
11 Pence by an 11-punch.

EDITING OF STERLING FIELDS

RPG provides for the editing of sterling fields, with columns 45-70 of the Output-Format Specification sheet containing the edit words. Each edit word is composed of three portions, or fields: the pounds field, the shillings field, and the pence field. The RPG program enables the user to make use of the following:

1. Floating and fixed pound signs.
2. Zero suppression of the pounds, shillings, and pence fields, or in any combination.
3. CR and minus (-) symbols.
4. Asterisk protection.
5. An ampersand causes the insertion of a blank in the edit word.

Rules for Forming Edit Words for a Sterling Field

1. An edit word must be enclosed by a set of single-quote symbols.
2. At least one separator character, or delimiter, must be inserted between the pounds and shillings fields and the shillings and pence fields. Any character can be used as a delimiter except an asterisk (*), a zero (0), a decimal point (.), and a comma (,). However, a comma is permitted within the pounds field, and a decimal point is acceptable within the pence field.
3. Asterisk protection if desired must be specified in the pounds field. This provides asterisk protection in both the pounds field and the shillings field.
4. When specifying the floating pound sign, there must be at least one pounds field position preceding the shillings field and following the pound sign.
5. Zero suppression occurs in the pounds, shillings, and pence fields. If no zero is encountered in the field, the

entire field is zero suppressed. Because the fields are edited separately, when a low-order zero is desired in a particular field, a zero must be placed in the rightmost position of that field where zero suppression is to stop.

6. No constant information can appear in an edit word.

7. The edit word must always be at least 2 positions larger than the input field on which the editing is to be performed.

8. The number of blank spaces supplied in the pence field of the edit word must be 2 greater than the number of decimal positions specified in column 52 of the Input or Calculation Specification sheets.

9. There must always be at least 2 positions allowed for the shillings field in every edit word specification.

10. When zero suppression is to be performed on the pence field, the user may wish to have the delimiter blanked out if the pence field is zero. To do this, the pence sign (d) must occupy the position in the edit word immediately following the two blank positions in the pence field.

The following examples illustrate valid edit words for sterling fields. £ denotes the pound sign, s the shilling sign, and d the pence sign. The output data to be edited is 005070L.

Edit Word	Result of Editing on Printout
No Editing	005 _ 7 _ L
Zero Suppression (No Editing)	5 _ 7 _ 3
b £ 0 b / 0 b / 0 b CR	£ 5 / _ 7 / _ 3CR
b b £ 0 & b b & b b -	£ 5 _ _ 7 _ _ 3 -
£b b b & b b S b b d -	£ _ 5 _ _ 7S _ 3d -
* b 0 b / b b S b b d	* * 5 / * 7S _ 3d
b £ 0 b : b b : b b & . d	£ 5 : _ 7 : _ 3 _ . d

APPENDIX D: CONVERSION ROUTINE OPERATION CODES

The following list shows the relationship between conversion routine operation codes and how they are specified.

Code	Factor 1	Factor 2	Result Field	Specifications That May Follow This Entry
EXTCV	Reference name	Name or Label of the user's routine.	Label of the field which will contain the track address of the record.	KEYCV
KEYCV	None	None	Label of the field which will contain the key of the record to be located.	Any other RPG calculations.

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

READER'S COMMENT FORM

IBM System/360 Basic Operating System
Report Program Generator
Specifications

C24-3387-3

- Your comments, accompanied by answers to the following questions, help us produce better publications for your use. If your answer to a question is "No" or requires qualification, please explain in the space provided below. All comments will be handled on a non-confidential basis. Copies of this and other IBM publications can be obtained through IBM Branch Offices.

	Yes	No
● Does this publication meet your needs?	□	□
● Did you find the material:		
Easy to read and understand?	□	□
Organized for convenient use?	□	□
Complete?	□	□
Well illustrated?	□	□
Written for your technical level?	□	□

- What is your occupation?____________________
- How do you use this publication?

As an introduction to the subject?	□	As an instructor in a class?	□
For advanced knowledge of the subject?	□	As a student in a class?	□
For information about operating procedures?	□	As a reference manual?	□

Other____________________

- Please give specific page and line references with your comments when appropriate. If you wish a reply, be sure to include your name and address.

COMMENTS:

- Thank you for your cooperation. No postage necessary if mailed in the U. S. A.

C24-3387-3

Staple

Fold

Fold

FIRST CLASS
PERMIT NO. 170
ENDICOTT, N. Y.

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
P. O. Box 6
Endicott, N. Y. 13760

Attention: Programming Publications, Dept. 157

Fold

Fold

Cut Along Line

IBM S360 Printed in U. S. A. C24-3387-3

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

Additional Comments: