GC21-7514-2 File No. S3-28

Systems

1 0 V

K 11

i

Introduction to RPG II Report Writing



GC21-7514-2 File No. S3-28

Systems

Introduction to RPG II Report Writing

Third Edition (July 1978)

This is a major revision of, and obsoletes, GC21-7514-1. Major changes include using the latest level specification forms and updating the terminology.

Changes are periodically made to the information herein; before using this publication in connection with the operation of IBM systems, be sure you have the latest edition and any technical newsletters.

This publication contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental. Use this publication only for the purpose stated in the *Preface*.

Publications are not stocked at the address below. Requests for copies of IBM publications and technical information about the systems should be made to your local IBM representative or to the branch office serving your locality.

This publication could contain technical inaccuracies or typographical errors. Use the Reader's Comment Form at the back of this publication to make comments about this publication. If the form has been removed, address your comments to IBM Corporation, Publications, Department 245, Rochester, Minnesota 55901. IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the the information you supply.

© Copγright International Business Machines Corporation 1971, 1978

This book is intended for persons who want to learn to write programs which will produce printed reports using the RPG II language. The book is designed to be used by a person with no previous knowledge of computers and programming, or by a person who already knows a programming language but wants to learn about report writing in RPG II.

After reading this book, you should not expect to be able to write complex RPG II programs. You will be introduced to only a small part of RPG II. Many important features in RPG II are not discussed in this manual. The RPG II reference manual for your system contains all of the information about RPG II features on your system. You should have gained enough background knowledge so that you can readily learn more detailed information—either from reference manuals, classes, or IBM personnel—which is required for writing programs for your system.

How This Manual Is Organized

The first chapter describes, in general terms, how a system operates and the things you must do to run a particular job. The information presented should answer such questions as:

- · What are the parts of a data processing system?
- · What is a program?
- · What is a programming language?
- · What is an RPG II program?
- · How is an RPG II program run on a system?

The second chapter describes the RPG II program cycle and the RPG II specifications you must write to do a particular task. The material in this chapter is arranged to provide a gradual development of concepts, proceeding from the simple to the more complex. Thus, it is important to read the material in sequence. Sample jobs are used to illustrate the concepts presented.

The third chapter explains an RPG II programmer's job more fully. It shows, by means of a sample job, the things a programmer must do from the start of a job to its completion.

RPG II Coding Forms

Following is a list of forms used to code and debug RPG II programs. Contact your local IBM branch office to order any of these materials:

- RPG Control and File Description Specifications, GX21-9092
- RPG Input Specifications, GX21-9094
- RPG Calculation Specifications, GX21-9093
- RPG Output Specifications, GX21-9090
- RPG Indicator Summary, GX21-9095
- RPG Debugging Template, GX21-9129
- Printer Spacing Chart, X20-1776
- Record Layout Form, X21-9088
- Disk File Layout Chart, GX21-9108
- Printer/Display Layout, GX21-9174

Contents

CHAPTER 1. BASIC DATA PROCESSING AND PROGRAMMING CONCEPTS	. 1
Parts of a Data Processing System	. 1
Input Devices	. 1
Output Devices	1
Processing Unit	1
Programs and Programming Languages	. , 2
Programs and Programming Languages	. J 2
Source Programs	. J E
Source Programs to Object Programs	. D
Summary	. 0
Data Processing Terms and Programming Aids	. /
CHAPTER 2. RPG II PROGRAMMING LANGUAGE	11
RPG II PROGRAM CYCLE	11
WRITING SPECIFICATIONS FOR INPUT AND	
OUTPUT OPERATIONS	14
Program Cycle Operations	14
Describing the Files	16
File Names	17
Device Designation	18
File Use	18
File Designation	18
Record Size	19
Describing Input Records	20
File Names	21
Field Names	22
Field Location	24
	24
	24
	20
	20
Record Type	27
Field Names	29
Field Location	29
Printed Reports	30
Spacing	30
Skipping	31
Editing	33
Job 1 (TRNREG): Printing a Simple Report Using the	
Three Basic Cycle Operations	36
Job Definition	36
Job Requirements	37
Job Specifications	38
	40
Brogram Cycle Operations	40
Program Cycle Operations	40
	42
	42
Describing the Result Field	45
Result Field Length	4/
Decimal Positions	4/
Half-Adjusting Results (Rounding)	47
Job 2 (TRNREG): Doing Simple Calculations	49
Job Definition	49
Job Requirements	50
Job Specifications	51
WRITING SPECIFICATIONS FOR INDICATORS	53
Control Level Indicators	54
Program Cycle Operations	56

RPG II Specifications	57
Using the Blank-After Specification	59
Job 3 (TRNREG): Using Control Level Indicators to	
Calculate and Print Totals	60
Job Definition	60
Job Requirements	61
lob Specifications	62
First Page Indicator	64
Program Cycle Operations	65
PDC II Specifications	66
	68
Bregreen Cycle Operations	70
	72
Heing Specifications	72
Using Spacing with Overnow	74
Using Overnow and TP Indicators Together	74
Last Record (LR) Indicator	75
Program Cycle Operations	75
RPG II Specifications	/0
Job 4 (TRNREG): Using First Page, Overflow, and	
Last Record Indicators to Print	
Headings and Totals	77
Job Definition	//
Job Requirements	//
Job Specifications	/9
Record Identifying Indicators	81
Program Cycle Operations	82
RPG II Specifications	83
Specifying Record Identification Codes	85
Specifying Record Identifying Indicators	86
Specifying Record Type Sequence	87
Job 5 (STOKST): Using Record Identifying Indicators	
to Process Different Record Types	89
Job Definition	89
Job Requirements	89
Job Specifications	92
Resulting Indicators	96
Program Cycle Operations	96
RPG II Specifications	98
Using the Compare Operation	99
Using an Arithmetic Operation	102
Job 6 (STOKST): Using Resulting Indicators to	
Test Contents of Result Fields	105
Job Definition	105
Job Requirements	105
Job Specifications	108
Field Indicators	111
Program Cycle Operations	111
RPG II Specifications	112
Job 7 (AGETB): Using Field Indicators to	
Test Contents of Input Fields	114
Job Definition	114
Job Bequirements	114
lob Specifications	116
Conditioning Operations by More Than	
conditioning operations by more man	
One Indicator	118

CHAPTER 3. THE PROGRAMMER'S JO	ЭΒ						121
Determining the Job Requirements							122
Determine RPG II Specifications Needed t	for	the	Jo	b			127
Write the Specifications				-	÷		129
Document the Program					÷	ċ	133
Prepare for Compilation					÷	•	134
Specifications Form Order					•	•	134
Control Specifications Preparation					·	•	135
Checking the Specifications				·	•	·	135
Compile the Source Program		•	• •	•	•	•	137
Test the Program			· ·				140
GLOSSARY			• •				141
INDEX							145

Chapter 1. Basic Data Processing and Programming Concepts

PARTS OF A DATA PROCESSING SYSTEM

Data processing systems differ widely in appearance, usually consisting of several connected units. Regardless of their size or shape, however, all data processing systems have common features.

Data processing systems operate electronically. They are complex machines with thousands of circuits and wires. Fortunately, you do not have to understand the purpose of each circuit or wire. To write a program which will produce a report, all you need to know are the purposes of the main parts of a data processing system: the input devices, the output devices, and the processing unit.

Input Devices

Data you give a system to work with is called *input*. The device used for getting that data into the system is called an *input device*. Several kinds of input devices may be used; among them are card readers, disks, diskettes, tapes, and keyboards. For the purposes of this manual, specific input devices will not be discussed. Information about the input devices applicable to your system is in the RPG II reference manual for your system.

Output Devices

Data produced by a system is called *output*. The device that produces the output is called an *output device*. Several kinds of output devices may be used; among them are card punches, disks, diskettes, tapes, printers, and display screens. For the purposes of this manual, specific output devices will not be discussed. Information about the output devices applicable to your system is in the RPG II reference manual for your system. Because this manual deals only with report writing in RPG II, we will refer only to printed output.

Processing Unit

The main part of a data processing system is the processing unit. The processing unit can be divided into three sections—storage, control, and arithmetic/logic-according to the special function each performs.

Storage

Storage is the system's memory area. This area is divided into many storage positions which the system uses to electronically store information. The actual number of positions in storage depends upon the size of the storage unit. Each storage position has an identifying number called an *address*.

A storage address serves the same purpose as a house address. Information is sent to and received from these locations. The information can be easily retrieved using the address where the information is stored.

Arithmetic/Logic

Calculations (such as add, subtract, multiply, and divide) are performed in the arithmetic/logic unit. When your instructions tell the system to do an operation such as add, the information to be added is transferred from storage to the arithmetic/logic unit. The operation is then performed and the result is sent back to storage.

Control

The control section is the system's decision maker. It retrieves instructions from storage, determines what has to be done, and directs other units or devices to perform the required operations.

PROGRAMS AND PROGRAMMING LANGUAGES

Data processing systems do only what you tell them. When you give a system instructions, however, it might seem as though the system requires more than you would need to do the same job. But remember, a data processing system cannot think: it requires explicit instructions, even for those things you would do almost without thinking.

When you are to do a job yourself, you need three basic things:

- · Information to work with (input).
- · Instructions telling you how to work with (process) the information.
- · Additional instructions describing the expected results (output).

In data processing terms, input is what you put into the system, processing is what the system does with the input, and output is the result of processing. Every job you run on a system has these three parts. You write instructions to describe what you want the system to do with each part. These instructions are called a *program*.

To communicate with the system, you must use the system's language, or one that can be translated into that language. The system's language is called *machine language*. It consists of letters, numbers, and symbols that, when properly arranged, have a specific meaning to the system and, when interpreted by the system, cause it to perform a desired function.

Because machine language is so very different from our own language, it is extremely difficult to use it to write a program. For this reason, programming languages have been created. A programming language allows the programmer to use familiar words and symbols to write instructions.

The RPG II programming language is composed of letters, numbers, and symbols which you put together to form an instruction (express a thought). When creating instructions in the RPG II language, you must follow certain rules just as you would when constructing a sentence in English. You will learn about these rules in the second part of this manual.

The set of instructions you write is called a *source program*. This source program is translated by a program called the compiler, resulting in a machine language program called the *object program*. It is the object program that you use to do a job. In fact, you can use it over and over to do the same job.

Source Programs

The instructions you write for any program must describe the input, processing, and output requirements of the job. For example, one instruction might direct the system to read an input record, another might specify the adding of two numbers, and another may tell the system to print a line on the printer. Because all jobs are not the same, you provide a different set of instructions (program) for each job. To write the instructions, you fill out RPG II specification forms (see Figure 1). These forms have been specially designed to help you write instructions according to the rules of RPG II language. The act of writing instructions on these forms is called *coding*; the entries you make on the forms are called *specifications*.

TDM		RPG OUTPUT	SPECIFICATIONS	GX 21 9090 UM/050					
LOT International Business Machines Co	rporation			Protect of U.S.A					
Program Programmer	Date	Keying Graphic Instruction Key	Card Electro Number	1 2 75 76 77 78 79 80 Page					
RPG CALCULATION SPECIFICATIONS GX21 9093 UM-050" Protect on U.S.A.									
Descourse Conternational Business Machines Con	rporation		· · · · · · · · · · · · · · · · · · ·	1 2 76 76 77 79 70 90					
Programmer	Date	Keying Graphic Instruction Key	Card Electro Number	Page d Program					
RPG INPUT SPECIFICATIONS GX21 9094 UM:050* IBM International Business Machines Corporation Printed in U.S.A.									
Program		Keying Graphic	Card Electro Number	1 2 75 76 77 78 79 80					
Programmer	Date	Instruction Key		Page of Identification					
			······································						
IBM. International Business Machines Cor	rporation	CONTROL AND FILE DE	SCRIPTION SPECIFICATION	S GX21 9092 UM/050* Printed in U.S.A.					
Program Programmer	Date	Keying Graphic Instruction Key	Card Electro Number	1 2 75 76 77 78 79 80 Page					
		Control Spe	cifications						
Size to Line 2 4 5 6 7 8 9 6 7 1 11 12 12 12 0 1 H	N. mbb By By By By By By By By By By By By By	er Reserved 28 27 28 29 30 31 32 33 34 35 36 37 38 35 File Description \$	A Sign Handling A Sign Handling B Sign Handling B Fight Translition B North MFCU Zuror B North MFCU Zuror B North MFCU Zuror B North MFCU Zuror B Sund I/O B Sund I/O B Fight Part of Hart B Sund I/O B Fight Part of Hart B Fight Part of Hart B Fight Part of Curor B Fight Curor <th< td=""><td>Refer to the specific System Reference Library manual for actual entries. 5 </td></th<>	Refer to the specific System Reference Library manual for actual entries. 5					
F Line 5 2 4 5 6 7 8 9 10 11 12 13 16	File Designation File Designation End of File Sequence File Format Block Length 1612 July 10 41 20 21 22 20 44	Mode of Processing Langth of Key Field or of Record Address Field Record Address Type Type Type of File Organization Nor Additional Area	Device Symbolic Device	Name of Label Exit File Addition/Unordered Storage Index File Addition/Unordered Continuation Lines File Addition/Unordered Option Entry					
0 2 F			1 TR. 27 TE TA TO	19 00 97 90 59 180 91 92 b4 96 185 66 67 68 69 79 71 72 73 74					
0 3 F			╋╦╋╋╋╋╋╋╋╋						
0 4 F			╊╫╊╊╊╊╋╉╡┼┼┤┤┝╊╉┊	┼┼┼┼┾╋╋╍╋╍╋╍╋╍╋┙					
0 5 F			┟╴╅╴╋╴╋╴╋╴╋╴┥╴┼╴┥╴┥╴╄╼┫╴╽						
0 6 F			╊ ╹╋╹╋╹╋╹╋╹╋╹╋╹╋╹┥ ╵┼╶┼╌╋╼┫╺╿	┼┼┼┼┾╆╆╋╋╋╋╋╋╋╋╋╋					
0 7 F			╊╌╋╼╋╌┥	┤ <u>╴</u> ┤ <u>╷</u> ┽ <u>┼</u> <u>╊</u> ┿ <mark>╋┉╋┉╋┉╋┙╋┉┫</mark> ╶┼╴ <mark>┣╍╉</mark> ╶┼╴┝╼╋╼┨					
0 8 F				┼┼┼┼┾╋╋╋╋╋╋╋╋╋╋╋╋					
0 9 F									
10 F									
	09 69 85 (3 95 99 90 70 70 70								
			. se re ce ce re ou de de 16 96 96 96 96 86 13 0	и огвіті Агді мі Єгії (1) (8 Т 8 д 8 С 1					

Figure 1. RPG II Specifications Forms

To describe the input, processing, and output requirements of your job, you supply different information on each form. For example, you have to describe what your input data is like and specify the device (such as a disk unit) that will read it. You also have to describe how the input data is to be processed. This includes specifying what type of operations (such as add or subtract) must be performed upon the data. Finally you specify what kind of output you want (such as a printed report), what information must be included in the output, and how that information should be arranged.

After you have coded the specifications forms, the next step is to get the coded information into the system. The system cannot read the coded forms, so you must put the specifications into a format that the system can read. Depending on your system, you key the specifications onto a diskette, into punched cards, or directly into the system.

Source Programs to Object Programs

As we said earlier, the system understands only machine language. It cannot use a coded program (written in a programming language like RPG II) directly. Any program you write in RPG II must be translated into machine language. The translator is a program called a *compiler*. The RPG II Compiler program is available from IBM.

The compiler translates your RPG II specifications (source program) into machine language (object program). The translating it does is called *compilation*. Essentially, the compiler performs three functions during compilation:

- It determines what machine instructions are necessary for the system to perform the job described by your RPG II specifications.
- It translates your RPG II specifications into a machine language program.
- It assigns storage locations to program instructions and data.

Summary

Figure 2 illustrates the RPG II data processing concepts.



Figure 2 (Part 1 of 2). Summary of RPG II Data Processing Concepts



Figure 2 (Part 2 of 2). Summary of RPG II Data Processing Concepts

Basic Data Processing and Programming Concepts 7

DATA PROCESSING TERMS AND PROGRAMMING AIDS

In the discussion of RPG II, you will find reference to data processing terms and programming aids. Figure 3 describes some basic programming terms. Figure 4 and Figure 5 are examples of some programming aids.



Figure 3. Basic Programming Terms

IBM

PROPORTIONAL RECORD LAYOUT FOR

Application			Type of F	Records	ITEM	TRANSP	ACTION	RECOM	RD	By	
RECORD NAME AND REMARK	01 05	OF COA	0B 0F	10 14	15 19	1A 1E	1F 23	24 2	8 29 21	D 2E 32	2 33 37 38
Uec.	1 5	6 10	ncler M	16 20	21 25 FM	26 30	31 35	36 4	0 41 4	5 46 51 1 TST	0 51 55 56
	Order	N	umberg	NUNN NL	MBER	ITEM	DESC	RIPTIC	₩	PRICE	PRICE
	+****			+	+++++++++++++++++++++++++++++++++++++++			+ • • • •		+	+++++++++++++++++++++++++++++++++++++++
			Lui	1				1	1	1	
	ſ			I			I	1		I	I I
	+		+	+	+	+	+	+ • • • •	+	+	+
			F I		1		ł			1	· ·
	╡┙╹╵╹╹┥	1111		+••••	+ • • • •	┼┶┷┶┶		$+ \cdot \cdot \cdot$	+++++++++++++++++++++++++++++++++++++++	+ • • • •	┼╹╹╹╹╹╹
					1		1		1	1	, 1 ,
	****	<u> </u>		++++++		+ • • • •	<u>↓ ↓ ↓ ↓ ↓</u>	++++++	+ • • • •	+++++++++++++++++++++++++++++++++++++++	+++++++++++++++++++++++++++++++++++++++
								1	1	1	1
	T		T	T		T		1	1	1	1
			ļ	<u> </u>	<u> </u>	<u>huu</u>		ļ	Luu	+	+++++++
			1	ſ	1			1		4	
	╡┥╹╹╹┥		$\frac{1}{1}$	┥╸╸╸╸	+	 • • • • •	+ • • • •	$+ \cdots +$	++++++	+	┽╍╍╺┽┶
			i t		1		1		1		• •
The record layout form	shows wh	hat reco	<u>liii</u> ards in a	<u>Li i i i</u> file look	lika Th	<u>Li i i i</u> is form i	<u>l i i i i</u> s filled c	<u>liiii</u> Nutiat		+	╅╍╾╸
the time a file is designed	ed. It sho	ws wha	t fields a	re in the	record ($\widehat{1}$ and the third of the temperature of tem	he exact	loca-		1	1
tion and length of each	(2) . It r	may als	o show f	ield nam	es and ex	kplain wł	nat kind	of data		1	1
is in each field (3) .										<u> </u>	<u> </u>
There are different reco	rd lavout	forms	for disk	80-colur	nn cards	. 96-colu	mn card	s. and		I	
tape. The form shown	above is th	he Prop	ortional	Record	Layout f	orm.		-,	<u> </u>	+	┥╍╍┍ ╁┹
	1				1						1 .
· · · · · · · · · · · · · · · · · · ·	+	1111	$\frac{1}{1}$	+	+ ++++ +	+		$+\cdots$	+	+	┥╍╍╺ ╋╸
	.		1		1			1.	1		i L
	++++++				+ *	+ + + + + +		+••••		+++++++++++++++++++++++++++++++++++++++	+
L					Lun	<u></u>	لىدىت	4	1	4	<u> </u>
• Hex.	00 04	05 05	0A 01	OF PC	314 18	19 10	1E 22	23 2	7 28 2	C 2D 3	1 32 36 37
Dec.	10 4	5 9	10 14	4[15] 19	ין∠∪ 24	125 29	1.30 34	4.55 <u>3</u>	alan a	41 °C 49	J DU 54 55

*Two numbering arrangements, each in hexadecimal and decimal notation, are shown. Select the arrangement and notation used by checking the appropriate box to the left. †The number of forms per pad may vary slightly.

Figure 4. Record Layout Form

ROG. ID	PAGE DATE		
.IST:		-	(4) (A)
		4 4 4 4 4 4 3 5 5 5 3 5 5 5 5 5 5 6 6 6 6 6 6 6 6 6	
$\begin{array}{c} & & & \\ & & & \\ & & & \\ & & & \\ & & & \\ & & H & 9 \\ & & H & 9 \\ & & & & \\ & & & H & 9 \\ & & & & & \\ & & & & \\ & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ &$	CUSTOMER MAME XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	RECEIPTS REGISTER INVOICE CAS NO RECEI XX XXXXXX XX XX, XX, XXX	H PISCOUNT AIR I VEP TAKEN CREPIT XX XXXXX XX,XXXXXX COISC (CREDIT)
13 16 17 7 18 19 20 21 22 23	s S		`© .xx* X.xxx.xx* xxx,xxx .xx* H) (1Ф. 21\$С)(1Ф СВЕД)
24 25 26 27 28 29 30 29 30 20 20 20 20 20 20 20 20 20 2	2 <i>Thiple</i> pacing chart indicates visionly ded in the recent	vhat a printed report should look li	ike by showing what
31 many differe 32 many differe 33 information 34 You determi 36 numbers corr 37 in spaces cor	nt lines are needed (1) to include in the lines (ne where to place infor respond to print position responding to the desir	, what spacing is required between 3, and the exact location of that mation by the numbers at the top of ons on the printer. Any heading inf ed print positions 5 Xs are used	nized. It shows how lines (2), what information (4). of the form. These formation is printed
39 data is to be 40 shown (1). 41 also use the F 43 report.)	printed (6) . Field nai Any punctuation to be Printer/Display Layout	mes are entered under the Xs to inde e used is also indicated on the chart Form, GX21-9174, to show the for	licate what field is (You could rmat of a printed

Fold back of dotted line.

Figure 5. Printer Spacing Chart

RPG II Program Cycle

When you do any job, you must do it in a particular order. The system must also do its job in a particular order. This logical order for the job is supplied by the RPG II compiler and/or your coding.

The logic the compiler supplies is called a *program cycle* (see Figure 6). The object program goes through this cycle of operations every time a record is processed. Depending on your specifications, the object program may or may not use a particular operation in the cycle. However, the program still goes through the complete program cycle every time. Because one program cycle is needed for each record read, many program cycles are required for every job.



Notes:

- 1. The program cycle shown gives the general order of the operations. There may be minor variations between this cycle and the detailed cycle discussed in the reference manual applicable to your system.
- 2. You do not need to memorize the program cycle. The cycle is only shown at this time to give you an idea of the cycle of the operations. The operations will be discussed in greater detail later.



It is important that you know the order of the operations in the RPG II program cycle. This enables you to write specifications that will make correct use of the cycle. By knowing the order in which the operations in the cycle are performed, you can organize your program in an efficient manner and save yourself unnecessary coding.

In this chapter, the operations in the RPG II program cycle are explained a few at a time. You will learn:

- · Which operations are used for a particular function.
- · Which RPG II specifications you must write to use the function.

Writing Specifications for Input and Output Operations

One of the simplest jobs you can do on a system is read information from an input record, such as a card, then put that same information out, such as in the form of a printed report. No calculations are done.

Program Cycle Operations

To do this simple job, the system uses only the three most basic operations in the RPG II program cycle. Figure 7 shows these operations.

Notice that two operations are concerned with the basic requirements of a job: input (read a record) and output (detail output). The third operation is the movement of data inside the system.



Figure 7. Three Basic Operations in the RPG II Program Cycle

Data read by an input device must be transferred to the system's processing unit before it can be used. Moving data is a mandatory operation done for every job. Because this operation is mandatory and is done exactly the same for every job, the compiler can supply instructions to do it.

When the program is executed, the program cycle is repeated over and over All three operations are used for every record in the input file. The term *detail output*, in the cycle operation, means that the specified output operations are performed for every input record.

It may seem strange that detail output comes before a record is read. This occurs, however, so that headings can be printed on a report. If a job (such as this one) does not print headings, no information is printed during the first cycle.

To make proper use of these cycle operations, your specifications must describe the records in the input file and specify how the output records should be created. You must also indicate what devices are used in the job.

DESCRIBING THE FILES

The File Description Specifications form is used to describe all the files used by your program. This information includes the name of the file, the device used with the file, and information on how the file is to be used. You fill out the indicated positions on the bottom half of the form.



You must describe, on a separate line, every file used in your job. Many simple jobs require only one input and one output file. In the first jobs we discuss, therefore, we will use only one input and one output file.

File Names

Every file used in a job must be named. The name provides you and the compiler a means of identifying the file. During compilation, the compiler associates the file name with other characteristics of the file. Thus, you can refer to that file by name throughout your program and the compiler knows exactly which file you are referring to.

The compiler, however, recognizes file names only if they conform to these rules:

- A file name must be 1-8 characters long.
- The first character of a file name must be alphabetic. (The letters A-Z and the @, \$, and # signs are considered alphabetic characters.) The remaining characters in the name can be either alphabetic or numeric.
- · Blanks must not appear between characters in the file name.
- No two files used in the same program can have the same name. (Because some RPG II compilers use only the first seven letters of an 8-letter file name, be certain, when using these systems, to make the first seven letters unique; for example, use TRANSACT and TRANFILE, not TRANFILA and TRANFILB.)
- The file name must begin in position 7 on the specifications form.



It is a good practice to assign meaningful file names: Meaningful names indicate something about the file, such as the type of records in the file or the use of the file. Because file names can be no longer than eight characters, abbreviations may be necessary. But these too can be meaningful. For example, the abbreviation CUSTCHG might be assigned to an input file consisting of records for all customers having charge accounts.

Device Designation

You must also specify which devices your job will use for input and output. The ones you use, of course, depend upon the system you have, the devices you have, and your job.

To indicate the device used for the file you named, enter the RPG II code name for that device in positions 40-60. The name must begin in position 40.

Note: In the examples in this manual, you will see shading in the device name positions rather than actual RPG II code names for devices. Code names differ for each system. The RPG II reference manual for your system will tell you the code names that are applicable.

During compilation, the compiler associates the file name with the device name. When you use the same file name in the rest of your program, the system will know which device to use.

File Use

You must also describe how each file and its associated device is used in a program. Files can be used as either input or output. (Files with other uses will not be discussed in this manual.) If records are read from a file, the file is an input file. If a new file is created during the job, the new file is an output file. Printed reports are the only output files discussed in this manual.

You specify file use by placing either an I (input) or O (output) in position 15:



File Designation

Position 16 is used to explain more about the use of input files. In this manual we are discussing the use of only one type of input file, the primary file. For this, you place a P in position 16. The designation of a file as a primary file in an RPG II program indicates that the RPG II program cycle will supply some of the program logic. For more information on primary files, and other types of input files, see the appropriate RPG II reference manual.

Record Size

When describing files, you must specify the length (in characters) of records in the file. Record length is entered in positions 24-27. Enter the length so the last digit is in position 27:



The record length specification does two things:

- It tells the compiler how much storage space to set aside for a record (input or output).
- It specifies how many characters must be read to get a complete input record.

Record size for card files is easy to determine. It is either 80 or 96 depending upon the size of cards you have. Maybe not all your cards have information in all columns, but all columns must be read to get an entire record. Blanks are placed in storage positions corresponding to unpunched card columns.

The size of records (lines) on the printer is also easy to determine. Printed records are limited by the size of your printer (the number of print positions in a line).

You may, if you wish, specify a record size smaller than actual printer size. If you do this, make certain that none of the lines to be printed are longer than the length you specify; otherwise the RPG II compiler will give you an error message which requires that the program be corrected and recompiled.

Records on other files, such as disk, can be any size. The maximum size is limited only by the capability of the device. When you use one of these files for a job, make sure you enter the correct record size; that is, enter the one established at the time the file was created

Other information may be required to describe how the input file is stored on the specific device in positions 29–38. See the RPG II reference manual for your system for these entries.

DESCRIBING INPUT RECORDS

Besides describing files, you must describe the records in each file. The compiler needs this information to create an object program that will read records properly. Input records are described on the Input Specifications form. Information needed to describe the record in a file includes the name of the file containing the record, the name of each field in the record, and where each field is located in the record. You fill in the indicated positions to describe fields and data in the record:



To help in describing input records, you can use the record layout form described in Chapter 1. This form shows the location and length of all fields in the record.

File Names

To tell the compiler which records you are describing, enter the name of the file containing them in positions 7-14. The name must be the same (and spelled exactly the same) as the one you assigned to the input file on the File Description Specifications form:



Field Names

To identify individual fields in the record, you must give each field a unique name. From information you placed on the File Description Specifications form the compiler determines the size of the storage area for each input record. The field names you supply on the Input Specifications form tell the compiler to divide this storage area into smaller sections so each can be addressed separately.

The rules for forming field names are as follows:

- · The field name must be from 1-6 characters long.
- The first character must be alphabetic. Remaining characters can be either alphabetic or numeric.
- · Blanks must not be placed between characters in a field name.
- · The field name must begin in position 53 on the Input Specifications form.



It is a good practice to assign meaningful field names. For example, a field containing customer numbers would be more meaningful if it were called CUSTNO rather than FIELDA. CUSTNO indicates something about the data in the field.

Enter field names one line below the file name, using a separate line for each field:



Be sure to name every field that contains information necessary for your job. If you need all fields on the record, name them all; if you need only a few, name only those you will need. The entire record is read, of course, regardless of how many fields you are using from that record. However, only information in the fields you name can be used by the program.

Field Location

After you assign a field name, you must tell where the field is located in the record. This enables the compiler to associate the field name with the right information. To describe the field location, you specify the position in the record at which the field begins and ends. Starting position is specified in positions 44-47 (From); ending position is specified in positions 48-51 (To). When a field is only one character long, starting and ending position entries are the same. Field location entries can be easily determined from the record layout form:



The compiler also determines field length from the To and From entries. The compiler needs field length to determine how many storage positions to allow for each field. If the field locations you specify indicate a field length of 6, the compiler allows six positions in storage for the field.

Type of Data

To complete your description of the input fields, the compiler checks position 52. This position indicates whether data in each field is alphameric or numeric. A numeric field can contain only numbers; an alphameric field can contain numbers, letters, and special characters.

If position 52 is blank, the compiler assumes the field is alphameric. For numeric fields, position 52 must contain an entry. This entry indicates the number of decimal positions (digits to the right of the decimal point) in the field.



Leave position 52 blank for alphameric fields. Enter 0-9 to indicate the number of decimal positions in a numeric field.

Although you do not include decimal points in fields in input records, you must consider them if you want correct calculation results and output data. By specifying to the compiler the number of decimal positions you know to be in a numeric field, you provide the information necessary to produce an object program that will handle numeric data with decimals.

Remember, any field used in an arithmetic operation (add, subtract, multiply, or divide) must be specified as numeric.

DESCRIBING OUTPUT RECORDS

Output records are described on the Output Specifications form. Information needed includes the name of the file containing the output record, the name of each field in the record, and the location where each field is to be placed in the record. You fill in the indicated positions to describe how the output records should look:



For a printed report, you make additional entries describing the format of the report; that is, make entries indicating the spacing and punctuation you want. These entries are discussed later under *Printed Reports*.

The printer spacing chart and record layout forms are useful when you are writing output specifications. The record layout form shows the organization of fields on a card or disk record; the printer spacing chart shows the format of printed records.

File Names

You indicate the output record you want created and the device you want to create the record by entering an output file name in positions 7-14. Make sure the name you enter is the same (and spelled exactly the same) as the name you entered on the File Description Specifications form for the output file:



Record Type

Three different types of records which can be specified on the Output Specifications form are heading, detail, and total. You usually find all three types in a printed report.

Heading Records	ACCOUNTS RECEIVABLE TRANSACTION REGISTER 07/11/								
	DATE	CUST NO	CUSTOMER NAME	JOURNAL NO	INVOICE NO	CASH AMOUNT	INVOICE AMOUNT	JOURNAL AMOUNT	
	07/11/	759820	SOUND OF THE SEVENTIE		063420		\$ 46.23		
	07/11/	633870	OLDE VILLAGE SHOPPE		063421		89.70		
	07/11/	642990	PARAGON TV SALES		063422		20.30		
	07/11/	122620	CANNIZONI STUDIOS		063422		129.76		
Detail	07/11/	682030	RAYMONDS RAPID REPAIR			\$ 63.80			
Records	07/11/	742950	SARATOGA VARIETY			29.72			
	07/11/	014280	BAKER BRADLEY & CO			43.50			
	07/11/	872060	UNIVERSITY ELECTRIC			97.75			
	07/11/	883290	VILLAGE MUSIC & TV	07-036				\$18.23CR	
\	07/11/	006280	ALLSTONS	07-037				10.70CR	
Record			TOTALS			\$234.77*	\$285.99*	\$28.93CR*	

Heading records are printed at the top of a page. They include report titles, column headings, or any other information needed to identify the kinds of information found in the report.

Detail records contain information about an individual item. Information in a detail record is often taken directly from an input record.

Total records are written after a group of detail records. They usually contain data that is the result of calculations on information in a group of detail records.

To specify record type, place an appropriate entry in position 15. If the printer spacing chart was properly filled out, you can refer to it to determine record type:

These entries specify the record type:


Field Names

To specify the information to be placed in each output record, you must name each field to be included. You specify these fields on separate lines of the Output Specifications form in positions 32-37. Begin the list of fields one line below the file name:



When listing the fields, make sure you enter a name that was previously given to a field (for example, a field named on the Input Specifications form). If the name you enter on the Output Specifications form is not one previously used, the compiler will not know what information you are referring to.

Field names are used to create the output record in the output storage area. Information is placed in the storage area one field at a time in the order you list them on the Output Specifications form.

Field Location

To specify where you want fields placed in the output records, you make an entry in positions 40-43 (End Position in Output Record). This entry must be the exact position where the last character in a field should be placed in the output storage area at the time the record is being created. The entry is easy to determine if you use a printer spacing chart or record layout form.



PRINTED REPORTS

When your output is a printed report, additional entries are needed on the Output Specifications form to make the report easy to read. Information must be neatly arranged in rows and columns with adequate space between items in a line and between lines.

Spacing

Your field location entries (positions 40-43) control space between fields, but to control spaces between lines you code positions 17-18 (Space).

You can have the printer single, double, or triple space between lines by entering the number 1, 2, or 3 in the appropriate positions. If you enter the number in Space Before (position 17), the printer spaces before printing the line; if you enter the number in Space After (position 18), the printer spaces after printing the line. You may enter numbers in both positions 17 and 18. By specifying three spaces before printing and three spaces after, you can have five blank lines between printed lines. The printer spaces three lines after printing a line, then spaces three lines before printing the next line. This next line is then printed six lines beyond the last line printed.



Skipping

You can use Skip entries in positions 19-22 to control spacing between lines on a page and to control printing the first and last lines on a page. A skip can be made before or after a line is printed. You indicate this by coding the skip in either positions 19-20 (Before) or 21-22 (After). The entry you place in these positions depends on the type of printer you have.

In this manual, we will discuss and show examples only for printers with tapeless carriage control. If your system uses a carriage control tape, you will find the appropriate entries for skipping in the RPG II reference manual for your system.

The Skip entry for a printer with tapeless carriage control can be any line number (1-112) on the printer paper. (The standard 11-inch printer paper has 66 lines per page when 6 lines are printed per inch.) A Skip entry in positions 19-20 indicates the line to which the printer skips before it prints the next line.



Printer paper is not rolled backward. If the printer is on line 50 when the program issues a skip instruction to line 10, the printer skips to line 10 on the next page.

Editing

Editing is a means of punctuating numeric fields by adding decimal points, commas, and negative signs. Editing can also suppress the printing of leading zeros (in the number 00149, 00 are called leading zeros).

When a numeric field is read into storage, it contains no decimal point or commas; when an unedited numeric field is printed, it appears exactly as it is in storage. A large number printed without commas or decimals is hard to read. Furthermore, an unedited field may not be meaningful when printed because of the way the system stores negative numbers.

The system uses the last digit in a numeric field to indicate sign (plus or minus). If the field is minus, the system combines a minus sign with the last digit. When a negative number is printed unedited, the combination of digit and sign appears as a letter. For example, minus 6439 prints as 643R. On the other hand, a positive field has no sign (a numeric field that does not have a negative sign is assumed to be positive). A positive field, therefore, prints normally. Positive 6439 prints as 6439.

The compiler can provide instructions to edit in a number of ways. All you have to do is enter an edit code in position 38 of the Output Specifications form. Many codes are available, each indicating a different type of editing. Figure 8 shows the codes and the editing done for each. Figure 9 shows some examples of editing.

Note: When you edit a field, you often add characters to it. When printed, the edited fields require more space than they did on input records or in storage. When specifying end position for an edited field, always take into account the spaces needed for the punctuation that will be added. The printer spacing chart shows the amount of space needed for the edited field.

	_		Sign For	Negati	ve Balance		
Edit Code	Commas	Decimal Point	No Sign	CR	– (Minus)	Zero Suppress	Print Out On Zero Balance
1	Yes	Yes	No Sign			Yes	.00 or 0
2	Yes	Yes	No Sign			Yes	Blanks
3		Yes	No Sign			Yes	.00 or 0
4		Yes	No Sign			Yes	Blanks
A	Yes	Yes		CR	-	Yes	.00 or 0
В	Yes	Yes		CR		Yes	Blanks
с		Yes		CR		Yes	.00 or 0
D		Yes		CR		Yes	Blanks
J	Yes	Yes			_	Yes	.00 or 0
к	Yes	Yes			—	Yes	Blanks
L		Yes				Yes	.00 or 0
М		Yes			_	Yes	Blanks
X ¹							
Y ²						Yes	
z ³						Yes	
¹ T ² T in	he X code rem he Y code is u a three to six nn/n	noves the plu sed for date digit field a	is sign of the fields. It su ccording to	e field. Ippress the fol	es only the le llowing patter	ftmost zero a n:	nd puts slashes

nn/nn

nn/nn/n

nn/nn/nn

³The Z code removes signs and suppresses zeros.

Note: The edit codes shown in the first column are used in position 38 of the Output Specifications form to punctuate the field named on the same line. Only numeric fields can be edited. The decimal point is automatically inserted in the correct position.

Figure 8. Edit Codes

F	ield Length nd Digits	1769532	02	00	000	041345
F C is	ield haracter- tics	Positive Number— Two Decimal Positions	Negative Number Two Decimal Positions	Zero-Two Decimal Positions	Zero-No Decimal Positions	Positive Number— Three Decimal Positions
	1	17,695.32	.02	.00	0	41.345
	2	17,695.32	.02			41.345
	3	17695.32	.02	.00	0	41.345
	4	17695.32	.02			41.345
	А	17,695.32	.02CR	.00	0	41.345
	В	17,695.32	.02CR			41,345
	С	17695.32	.02CR	.00	0	41.345
Edit Codes	D	17695.32	.02CR			41.345
	J	17,695.32	.02–	.00	0	41.345
	к	17,695.32	.02			41.345
	L	17695.32	.02–	.00	0	41.345
	м	17695.32	.02-			41.345
	x	1769532	ОК	00	000	041345
	Y	Must be used with	a 3 to 6-digit field.		0/0	4/13/45
	Z	1769532	2			41345

Note: This table shows the effect of editing on five different fields. It illustrates what will be printed by using each edit code on the fields.

Figure 9. Examples of Editing

JOB 1 (TRNREG): PRINTING A SIMPLE REPORT USING THE THREE BASIC CYCLE OPERATIONS

Job Definition

Print a report listing all items sold during a week. The selling of an item is known as a transaction, so the report is titled *Transaction Register*.

During the week, a transaction file is created. At the end of each day, transaction records are created from information obtained from order forms received during the day. To get the printed transaction report, you list the information from all input records on the printed report.

Job Requirements

Input: Sales transaction file consisting of 96-character records. The format of the input records is shown on this record layout form:



Output: A printed transaction register:

7/23/	413010	CHOUI BUX 100A FLUSH	10	4.90
7/23/	412146	CH148 BREAKER 15A	100	•89
7/23/	411116	1500 TWIN SOCKET B	500	1.12
7/24/	503029	MUTUR 1/2 HP 60 CYC	2	146.78
7/24/-	317802	TERMINAL CLIP	100	5.12
7/24/	326917	TERMINAL BAR	100	4.12
7/24/	411121	1506 SOLKT ADAPT BRN	400	.19
1/24/	412997	CH173 BREAKER 3JA	60	1.15
7/24/	413088	CH176 PREAKER 60A	40	1.15
7/24/	+11174	C151 SIL SWITCH BRN	200	1.16
7/24/	413790	CHOUS BR BOX 150A	10	4.98
1/24/	718326	FCBU3 FUSE 15A	200	• 32

This printer spacing chart shows how the report is formatted:

	-			_	-		_																																																										
		1 2	3	• 5	6	, .	• 0	<u>,</u>	1123	1	31	5 7	1		2	22	2	2	2	2 2	2 2	3	3	3	34	3	3 3	3	3		4	4	4 4	4	4	4	5	5	5	5	5 5	2	5	6 6	6	6 6	6 6	6	5 6	7 1	7	77	7	77	7	7 8		8 8					9 9		9
	5	Ť	††	11	H	+	+	Ħ	+	Η	+	+	H	1	t	H	+	f	-	Ŧ	+	+-	Ť	+-	Ĥ	-	+	-	Ť	Ť	+			ľ	Ť	+	Ť	+		+	+	ł ł	+	Ť	+-	+		11		-	-	7	1-1-	+	₽	-	++		17	<u> </u>	4.		4	H	-
	2	+	++	1-	-t	++	+	H	+	H	++	ϯ	t t	+	╋	╉┥	+	╈	┝┥	+	┿	┼┨	╉	+	+	+	+-	+	H	╉	÷	┝┥	÷	┞┥	+	+	+	+	┽┥	+	+	<u>∔</u> +	+	+	┼┼	+	+	÷t	-	\mathbf{H}	++	-+-	+	+	⊢	+	╉┼	-+-	┢┼╋	++	+	+	+	╉╋	-
	-	+	++	+-	+	++	+	┢┼	+	H	+	+-	┥┦	+	╀		+	+-	┝┼	╉	+	H	+	+-	Η	+	+	H	+	+	+-	┝╺┝	+-	i +	-+	+ +	+	÷	+ i	+	+-	┥┥	+	┢╋	┿╋	+		++	+-	┝╍╋	++	+	++	+-	╀┥	+	╉┥	+-	┝┿	++	-+-	+		┿┿	-
		+	┿╋	+	+	╈	+	┢┼	+	┼┥	+	+	+ +	╈	t	H	+	╀	⊢	+	+	Η	+	╉	Н	÷	-+-	÷	H	╉	÷	┝	+	Η	+	+	+	+	H	+	+	┥┥	+	++	┿┿	+-	÷	÷ł	+-	÷₽	+		++	÷	+ +	+	╉┼	+	┽┼	+	┿	+	┝	╉╋	-
ħ		+	++	++	+	$^{++}$	+-	┢	\mathbf{t}	k	V.	$\overline{\mathbf{v}}$	V	+	+-	i ł	+	÷	h	\mathbf{x}	\mathbf{k}	Ы	d	+	Н	-+	╉	ti	-	t	ħv	ί,	$\dot{\mathbf{\lambda}}$	М	(Å	$\mathbf{\lambda}$	*	. א	\mathbf{k}	\mathbf{A}	$\mathbf{\lambda}$	~	\mathbf{k}	┢	┥┥	+	•	1	$\mathbf{\lambda}$	V	11	-10	A.	7	V	0	₽		┢┼┝	++	-+-	+	-	┿┽	-
-	- 4	+	++	++	+	++	+-	μ	Y	η	4	1	A;	+	+-	┝┼	÷	╀	Ăł	4	Ψ	И	4	+	┼┤	+	+	+	+	Ŧ	V	Ā	ΔA	А	4	Ψ	4	Ψ	Y	4	Ψ	₩	Ψ	4	+	-+-	4	ψ¥	Ψ	4	┽┼	<u>م</u>	44	4.	Α	<u>x</u>	₩	+-	┝┽	-4	-	+		┿	-
	-	+	┿╋	┥┥	+	┿	+	H	*	, 	+	-	h	+	╀	┝┥	+	+		_	╞	H	+	+	H	+	+	H	+	+	+7		+	H		+	5	\pm		+	÷	h	4.	H	$\frac{1}{1}$	+	ᡩ	낢				-	Li	+	++	1	6.	÷	⊬	++	∽	+	+	┿┿	-
		+	┿╋	+	+	++	+	μ۴	K 1	1	4	Ŧ,	4	+	┿	i ł	-4	μ	+4	f	4/	M	+	+	\vdash	+	+	⊢	-+	+	4	Ķ,	44	F4	M		4	4	Х	4	μ	r4	μ	24	4	+	-4	φ	4	4	H	4	ļ r .	Ψ.	i i	Ų	11	41	4	-4	·4	++	╋	┿┿	-
	-	+	┿╋	+-+	-+-	┽╋	÷	╉┽	+-	+	+	+-	łŧ	+	╀	H	÷	₩	U	*	t	9	4	R	₽.	-+	+	+-	+	╉	+	++	-+-	H	-+	+-+	+	+	++	+	÷	+-+	÷	+	H	-+-	+	÷+	+	+	++	-+	++-	-	+		╂┼	+ •	Η÷	-+-+	+	Ηİ		╉┥	_
	10	╋	┿╋	+-	+	+	+	╉┊	+	+	+	+-	H	+	┢		÷	+	i i	+	+	+	4	Ψ	4-1	+	+	+	+	+	+	÷-+	-+	H	+	+-+	+	+	++	÷	-+ -	ļł	4	+	+ +	+-	-+-	┽┼	+-		┿┿	-+	┝┿	-+	H	+	╉┼	+	┢╌┿	++		++		++	_
		-	┿╋		+	++	╈	╉┿	+	H	+	+	╉┥	+	┢	H	÷	+-	H	+	+.	+	+	+	H	+	+	+.		╁	+-	i-i	-+	+ +	+	+-+	-+	+	+1	+	+-	<u>+-</u> +	+-	+	i i	+-	+	+-+	+-	-	╉┥	-+-	++	+	∔∔	+	╉	-	++	++	+	++		++	-
		+	┿╋		+	╉	╉	╉┈┿	+-	┿┥	H	\$	H	1	b	H	ø	+-	H	¢	1	Ы	-	+		┝┼	+	•	A	4	+	١.,	h	1	>	4	+	÷	+	+	+	++	+-	+	+	+-	-+-	+	++		++	-	++	+-	┿┥	-+-	łł	+-	₩	-+-+	+	łł		++	_
		++-	++	44	H	++	+	╉┽	+	H		+	H	+	₽	Ľł	¥	+	┝┥	Ŧ	Ŧ	Н	4	Ŧ	F	4	f	7-	4	+	+	1	4		4	7	4	+	+-	-+-	+	ł	+	+	++	-i-	-	٠ł	+	4	łi			+	Ļ	÷	łł	4	44	┿┥	+	44	_	₩	-
		+	ŧŧ		┝┿	+	+	╉┼	╉	┝┤	-+-	+	+	+	┿┈	H	+	+	÷.+	+	+	+	+	+	┝	-+	+	+	\vdash	+	+	++	+	1	-+	+	+	+	+-	+	+-		+-	+	┼┼	÷		i.	+-	++	+	- -	++	+-		4	+	+	₩	+	-+-	+	+	++	_
	14	+	† †		\vdash	┽┥	+	╉┼	╉	┥┥	-+-	+	Hi	+	╀	┞┥	+	╀	+	+	+	+	+	+	ł	+ +	-	+	\vdash	+	4-	Ļ	÷	i	+	+	+	-	+-	+	+-	H	+	H	++	+-	+	++	+	4	+	+	∔∔		++	÷	╂∔	+-	⊢∔	-+-+	+	++	+	++	-
	13	H	++		┝╌┿	┽┥	+	╉┼	╉	Н	++	+	ł	+	╀	┝┥	+	+	┝╌┥	-	Ŧ	Н	4	+	+	-+	-	Ļ	-	+	+	Ļ,	+	-	+	+	-	+	+	+	+	++	+	+	H	+	⊢	₩	+-	++	\downarrow	- -	+÷	+	∔ ∳	+	₽₽	+	┿	++	-+-	ΗÌ		₩	_
	10	+	╉		+	++	÷	╉┽	+-		+	+	H	+	╀	┝┥	+	╀	H	-+-	+-	+-	-+-	+	H	\vdash	÷	+	1	+	ļ.	4			4	-	-	+	44	-	+	┿┅┿	+	-	\downarrow	+	-+-	++	+	┝╌╋╴	+ +	+-	++	+	+ +	+	++	+	÷÷	++	-	łi	-	++	-
<u> </u>	17	++	++		H	++	+	₽∔	+	H	+	+	ιį	+	∔	H	+	+	H	+	1		-	+	L		4	1		4	+	4		+-+	+	+	-	\downarrow	+	-	÷	14	4	4	+ +	+-	+	++	+	++	+ +	_÷_	++	┶	4	-÷-	4	+	₩	++	+	\downarrow		44	
	18	+	∔∔		-+	++	+	₩	+	μ	4	+	H	4	∔	Ц	+	+	4	+	1	\downarrow	-	\downarrow	H		\downarrow	Ļ.	\square	┛	1		4		4	+	4	+	-	4	4	44	+	4	44	+		44	+-	4		1	₊		i l	-	Ļ	+	4	4	-	4		4	
	19	4	++	44	\mid	+	-	₽	+	\downarrow	Ц.	+	Ļ	\downarrow	∔	H	-	1		+	1	Ц	4	1	L		-	1		┛	1	Ц	÷		1	1		1	1	-	1	$\downarrow\downarrow$	+	4	14	4		1		4	4	; 	44	\downarrow	1	+	Ц	-	4	1	4	\downarrow		$\downarrow\downarrow$	
	20	4	++	44	μ	++	4	\downarrow	+	1	\vdash	+	H	+	┢	\downarrow	-	+	Ц	_	+	\downarrow	4	+			+	÷		1	1	L	1		4			-	\downarrow	-	4	$\downarrow\downarrow$	1	4	Ļļ		4	44	4		\downarrow	4	44	4	ĻÌ	4	Ļį	\downarrow	H	\downarrow	+	44		$\downarrow \downarrow$	_
	21		++		H	\downarrow	+	\downarrow	_	1	4	\downarrow	H	4	∔	\downarrow	+	\perp	\downarrow	_	+	\downarrow	4	4	1	L		1		4	\downarrow		-		-			\downarrow	i		+	H	\downarrow		44	+		H		4		-	44	_	\square	4	H	+	4	\downarrow	\downarrow	4		$\downarrow\downarrow$	
			1 1	1	1 I.		- I -			1 1		+	1 1			1 1		1	1 1			1	- 1	1	1	1	1	1	1	1	1	11		1					1		1	1 1	F	I F	+ +	- 1	1	1.1			1	1	1 1	3	1 1		1	- 1	1 1	1.1	. 1	1.1		11	

Job Specifications



¹The Record Identifying Indicator (entry 01 in positions 19 and 20) is needed to get correct output. It will be explained later under *Job 5 (STOKST): Using Record Identifying Indicators to Process Different Record Types.*



¹ The Record Identifying Indicator (entry 01 in positions 24 and 25) is needed to get correct output. It will be explained later under *Job 5 (STOKST): Using Record Identifying Indicators to Process Different Record Types.*

Writing Specifications for Calculation Operations

Most jobs require some processing. In RPG II, processing can include calculating, comparing, moving, or changing data. In this discussion we will consider only calculating (adding, subtracting, multiplying, and dividing).

Program Cycle Operations

When you specify a calculation operation related to each input record processed, you are adding one more operation to the basic program cycle: the detail calculation operation (see Figure 10).



Note: This is a basic program cycle showing the addition of detail operations. Because this is a detail operation, it is performed during every cycle for every record read.

Figure 10. Program Cycle for Detail Calculations

The Calculation Specifications form is used to describe the operations you want performed. Information needed includes the type of operation to be done, the field(s) or constant to be used in the calculation, and the location where the result of the calculation is to be placed. You fill in the indicated positions:



Specify one operation per line. In each program cycle, processing steps are done in the order you specified on this form. If calculations must be done in a particular order, you must list the operations in that order.

DESCRIBING TYPE OF OPERATION

To indicate the type of calculation operation, you enter one of the following operation codes in positions 28-32 on the Calculation Specifications form:

ADD (add)

SUB (subtract)

MULT (multiply)

DIV (divide)

DESCRIBING DATA TO BE USED

After you have specified the type of operation, you must identify the data to be used. If you specified ADD, for example, you must tell the system what to add. You do this by naming the fields to be used in positions 18-27 (Factor 1) and 33-42 (Factor 2).

Instead of naming a field in Factor 1 or Factor 2, you can enter a *constant*; that is, you can enter actual data instead of the name of a field containing the data. For example, you can enter either of the following:

500 Constant (actual data)

AMOUNT Name of a field containing data

Constants can be either numeric or alphameric, but for now we will discuss only numeric constants. The rules for using numeric constants are as follows:

- · Constants can be up to ten numeric digits (0-9).
- Constants can have a sign and decimal point. The sign, if used, must be the leftmost character. The decimal point, if used, must be shown as part of the constant (4.12).
- The first character of the constant must be placed in the leftmost position of the Factor field.
- · Constants cannot contain blanks.

The contents of a field can change during execution of a program, but constants do not. If you want to add, multiply, subtract, or divide the same number during every program cycle, you can use a constant:



To the compiler, a constant is like a field name. During compilation, the compiler checks Factor 1 and Factor 2 for constants. If there are any, the compiler assigns a storage location for the constant and gives instructions to the computer to put the appropriate constant in that location at the beginning of job execution.

When you enter the fields in Factor 1 and Factor 2, be sure to consider their order because the specified operation may have an affect on the result:



DESCRIBING THE RESULT FIELD

You must specify where you want the result of a calculation stored by naming that field in positions 43-48 (Result Field). The name you enter in the Result Field can be the name of a field already defined on the input or calculation specifications forms or a new field.

You would not need to name a new result field in these two situations:

- The contents of an input field are no longer required in your program, and the field is the correct size.
- The contents of an input field or a result field already defined on the Calculation Specifications form is to be replaced with a new value.



If you name a new field, you must specify field length (positions 49-51) and decimal position (position 52) so the compiler can assign adequate storage for the new field:

Result Field Length

When you name a result field, make sure you specify one large enough to hold the results. Always consider the length of the fields involved in the operations. For example, if you are adding a two-position field to a three-position field, you must determine the largest result you could possibly have:

Because there are four digits in this result, you would specify at least 4 as the result field length.

If this calculation would occur many times in your program, as in a running total, you would probably need a result field length larger than 4. It is up to you to determine the largest field length needed; failure to specify a large enough result field can mean a loss of data.

Decimal Positions

For a new result field, be certain to place an entry in position 52. If the new field is to be numeric but contains no digits to the right of the decimal point, enter a zero. Remember, this entry indicates type of field (numeric or alphameric) as well as decimal position. If the result field is not specified as numeric by an entry in position 52, the compiler will not provide instructions for the calculation operation.

Half-Adjusting Results (Rounding)

In RPG II, rounding results is called *half-adjusting*. When the digit to the right of the last digit you want to keep is greater than 4, 1 is added to the last digit. The number 3.14159 rounded to four decimal positions becomes 3.1416. The same number rounded to two decimal positions is 3.14.

To half-adjust any calculation result, you place an H in position 53 of the Calculation Specifications form on the same line as the field to be half-adjusted:



In this example, DISCNT is half-adjusted. The entry in position 52 (Decimal Positions) indicates the number of digits to be retained after half-adjusting is completed. In this case, two digits are required. The multiplication and half-adjusting would be done like this:

- 74.98 Assumed value of PRICE.
- x .06 Constant representing 6% discount rate.
- 4.4988 Result which must be half-adjusted to 2 places.
 - 1 1 is added to 9 because 8 is greater than 4.
- 4.5088 Slashed digits are dropped since only two decimal positions are required.

Job Definition

Print a report listing all sales transactions for a week. This report is similar to the report created in Job 1. The only difference is the addition of the last column on the report which is the sales amount per item. Sales amount (quantity sold times item price) is not found in the input record and must, therefore, be calculated.

Job Requirements

Input: Sales transaction file consisting of 96-character records. The format of the input records is shown on this record layout form:



Output: A printed transaction register:

//:3/ ==	913 (1) 1 1 (CHOOL HUY IUDA FLUSH	10	4.90	49.00
	912146	(ΗΙ4Ρ ΡΚΕΑΚεΡ 15Δ	100	•89	89.00
/ .)/ =-	911115	1500 TWIN SUCKET B	500	.1.12	550.00
//4/	2,2050	MUTUR 172 HP 50 CYC	2	145.79	243.56
/ < 4/	317352	TERMINAL LLIP	100	5.12	512.0.1
1.4/	3267+7	TERMINAL BAR	100	4.12	412 00
/ 4/ ==	+11121	1506 SOCKI ADAPI BRV	400	.19	76.00
/24/	+1,977	CH173 PREAKER BUA	6.1	1 15	10.00
1041	413359	CH176 PREAKER AUA	6.0	1.15	54.00
1:4/	+11174	CIST STE SWITCH RUN	10.	1.1.7	45.00
/24/	+131171		200	1.10	232.00
1241	7183.25	6:0.3 E.C. 154	ιu	4.48	49 . 80
	110127	-CAUD -USE 15A	200	• 32	64.00

This printer spacing chart shows how the report is formatted:

		T	TT	Π	П	1		Τ,Τ	$\overline{1}$	5	IJ	<u>,</u>	1	$\begin{bmatrix} \\ \end{bmatrix}$	1	1.	J	T	Π	Τ	T	H	T	Π	Т	Т	Π	Т	Π	Т	TT	Т	ГТ	T	П	TΤ	Τ.	П	Т	Τ-	П	Т	Т	T	TT-	Τī	ТТ		ΤT	T			m	TΤ		тт	
		23	4 5	6 7	89	<u> i</u>	2 3	4	56	78	9	រំ	2 3	4	56	78	5	ð i	2	34	5	3 6	7	3	8	2	43	4 4	6	44	4	5	5 5	5 5	5 5	5	5 5	66	5 6	66	6 6	66	6	ZZ.	77	7	77	7 7	78	8	8 8	8 8	6 8		8 9	9	9
-+1	4			++	-++		÷÷	i	-		1	Γ	Π	1 :	T		1	1	T	1	1		Ť	11	+		ft	+	tt	+	tt	+	-				+	ľ.		+		++•	+++	Ŧ	2 3		기미	18	90	<u>"</u>	23	4 3	617		90	14	2
2	H		++	++									ΓT			÷	11	T	11	T	+-	H	1	11	1	Ť	<u>+</u> +	1	! †	+	**	t	++-	++	++	Ħ	-+-		┿╋	+	+	++	++	+	++	††	++	+	┝┾╌	₽∔	++	+	⊢	₩	+-	┢┼	+
3	↓.↓		++	+	4	+		Li	+ +			\Box				Ī	11	T	T	T	1		1	11	+	t	++	+	H	+	11	+	+	++	+	††	+-	++	+ +	╋	÷	┽╇	+	+	+-	┿╂	┽┽	+	i i	╂┼	++	+-	┝╌ᢤ╌	╉╋	+	┢┼┥	+
	11		++	44	++	+	⊢∔-	Ļ			+-∔	L	·	-+ 4		1	: T	Τ	Π		1			Ħ	T	+	,		+	+	! 	t	-+-	1	+	++	+	H	++	-+		• +-	$^{++}$	+	<u>⊦</u> +	┿┿	++	+-	┝┿	H	++	┽┥	r+	++	+	₽	+
D 6	┟┙	++	++	+	++	-	\downarrow	ł.	, ,	u vv	+	\downarrow	+-+-] +-†	-17	1			Д				Ţ	Ļ	Ţ	***				1	Ħ	İ	+				1		Ħ		+	;†	++		+	††	Ħ	+	++-	Ħ	+	++	+	╀╀	+	+	+
7	Ħ	-+-+-	++	Ť	t	-14	4	4	44	44	1+	┿	-+-	++	Ă	ĂΧ	(X)	¥Х	44	+	+ +	+	+	Ļ	₽	Ŷ	Σ_{2}	(X	X	¢X	Ø	K	XX	X	XΧ	\otimes	Ø	X			XX	X	X)	X		X	X	.χ	X	t b	X	.М	XX	1.5	ХX	It	t
	Ħ	++	++	11	-++		でわ	d.	÷	=h	٠ţ	++	÷	r^{\dagger}	#	-		tr		4		+	+	H	+	×.	_	-	4	*		H	-	i l	+	4	4.,		11			1				11	Ш	Τ		Π	П	Π	T	Π	T	Π	Ţ
9	Ħ	++	++	++	++	+-	¥	n	44	1	۰ŧ	++	-+-	ΨĄ	1,6	μų	μĄ	Ψ	η	đ٤	14	A	+-	i i	+	Ų,	Ľ.	¢Δ	И,	¥	ŧ٩	49	K	14	21	ĮΦ	₩,	4.	14	Q	UΡ	ŧΝ.	T	Έ.	Ŋ.	Ľ	PI	81	ĊĒ	IJ	10	\$A	14	₽₿	5	Π	T
10	Ħ	11	11	<u>††</u>	***	T:	-+-	+-+ .	+	- +	•	+		÷÷	- +	t÷-	++	+	łł	+-	+	-+-	+	┝╺┝	+	44	•	++	++-	∔.				i	-+-		Ļ	+	44	4,			$\downarrow\downarrow$	1	4.	1		1		Ľ	41	ĀĀ	241	h	± 1	\square	+
11	Π		T			T		ΓŤ	+-+	+-	++	14		1	\pm	1	++	╘	t t	t	H	+	+	ίļ	1	1	>	$\frac{1}{1}$	+	, La	+	H	-	ίł	+	ŧ.ŧ	+	+	i i	++	+	i -	\downarrow	+		H	44	÷-		11	11	11	1	[]	Ľ	Ĺ	
12	Π	T	Π	TT	11	Т	1		$\uparrow \uparrow$	+	•	11	41	+*+	4	14	$^{\dagger \dagger}$	P	14	1	¢.c.	<u> </u>	+-	4	Ψ	+	4	μ'n	Ę	\$	•	⊬	÷	┼┼	+-	+	++	-+-	╉╋	+ •	-	4	┢╌┢	+	-	4	++	-		4	++	4	_	11		1	4
13			П	П	Π	Т			+ +	-	++-	Ħ	+	***	++	+-	*+	+	++	+	11	┿	╈	H	╈	Η		+	+	+-	+	+	+	+	+	H	+	+	÷+	++	+		∔∔	+	4.	H	+-+	+	-+-	H	\downarrow	4	4		4	Щ	Ļ
14			П	TI		T	1			+		$\uparrow \uparrow$	+	+++	++	+	<u>+</u> - <u>+</u> -	t	• •	┿	tt	+	+	┝┼	╉	H	+	++	+	+	++-	+	÷	÷+	+	∔+	+	+-	ł ł	+	+-	4-	÷+	+	+		++	+			++	++	+	μ	$\downarrow \downarrow$	H-	Ļ
15				IT		Γ	i			1		T	-	+++	د ۲	;+-	++	+	t t	-+	+ +	-+-	t	┞┼	╈	÷	++	+ +	÷	ł	+	+	t	+ +	+-	H	+	+	++	+-	+-	++-	┟┟	╇	++	i -	٠Ļ	4	4	H	++	-+-+	4		44	+	4
16	\square		Ш	Ϊİ	Π				11		-	11	+	+ + :	+++	, <u> </u>	++	t	ΙŤ	+-	++	+	+-	H	t	t		÷÷	+	ti	-+-	╉┥	·		+-	H	+-i	+	++	+	+-	+	$\left \right $	+	-+-	┝┢	++	+ 1			++	- <u>+</u> -+	\rightarrow	Ц.	++	+	Ļ
17					ii		T		i i		Ť	Ħ	-	Ħ	+++	-+	++	ϯ	Ħ	+	tt	÷	t	+	┢	t	-+-	++	+	+-	+	╉┥	+	┼┼	+-	++	+ +	+	H	+ i	+-	-	ł ł	╉┥	-+	i + -	i +	+	-+-		++	+	44	4	1.	-	Ļ
18	\square			i l		Е			ΤŤ	Τ		TT	1	H	-		11	1-		+	ţ†	+	┢	H	╈	† ł		$^{++}$	+	+	÷	┢┥	+	$^{++}$	+-	H	++	+	++	÷	+-	+	┝╌┝	łł	+		Η	+ 1	+-	+	++	++	-+-	+	++	+-	Ļ
19	i				Ш	Г			Π			T	1	++-	++		++	+	H	+	Ħ	+	ϯ	H	+	H	÷	+ +	+	+	÷	ŧŧ	+-	Η	+-	H	+	╋	┿╌┿╍	++	-	+	┿┿	╉┼	+		÷+	+	-+-	Ļ.	↓↓	++	++	4	+	+-	÷
20	4	+ +-	11	44	1				П	1		П	T		++	+	***		H	+-	tt	+	1	-+	t	+-+	+	11	+	+	+	╉┼	+	┝╋	+	+	++	+	÷+	┼┽	ŧ	+-	<u>+ +</u>	₩	+	++^	i+	+ +		+	÷+	┿┿	- i +		┼╂	-+-	Ļ
21	\mid	++	∔∔	4	44	1	-		I	1		Π	T	Π	+ + +		+	T	T	+	H	╈	t	+	t	Ħ	1	††	+	11	+	t	+	+	+-	H	┿┥	+	┼┼	+	+-	+	+-	╂┤	Ť	+	Η	+	+	H	┽┽	+	-++	+-	++	+	╀
22	\vdash	÷.	++-	11	++	-	į.	4	ļĮ			П	I					Π		1	Ħ	1	Ħ	+	1	tt	+	††	-+-	\dagger	+	11	+	++	+	┝╋╴	+ 1	+	<u>+</u> +	††	+	+-	++	╂┼	+	++	++	Ηİ	+1	╟┼	++	+	++	+	┼╂	+-	ł
23		-i-i-	44	: + +	\downarrow	++	4	1	4	\perp		\square		L.T.	\Box			Π				1	Π		T	Ħ	+	† †	+	++	+	tt	+-		+		††	+	++	++	+	+	\uparrow	╂┦	÷	-+-	+-+	+	+1	┝┼	╆╋	++	++	•	┼╀	+-	+
24		I		1			1.						1	I I	1	1		Π		Τ			Π	1	T		1	T†	1	11	1	Ħ	1	Ħ	\uparrow		††	T		11	\dagger	+		† †		+	++		+	+	╈╋	++	++	+	┼╂	+	ł

Job Specifications



RPG II Programming Language 51

CALCULATION SPECIFICATIONS RPG

GX21-9093 UM/050* Printed in U.S.A



Writing Specifications for Indicators

So far you have learned how to use an RPG II program cycle for producing simple reports. However, actual business reports would be more complex. They would include more information, have page and column headings, and probably include subtotals and final totals.

A report like that shown in Figure 11 would require printing four different types of lines: report heading, column headings, detail lines, and total lines. Some of these lines must be printed only at certain times: headings would be printed only at the top of the page and totals only after all detail lines are printed. To produce the report correctly, you must use *indicators* to specify when you want certain things done.

To you, indicators are two numbers or alphabetic characters you specify on the specification forms. To the object program, indicators are like switches, located in the system. They mean one thing when on; another when off. You can use several types of indicators; each type signals something different.

You must know how program cycle operations affect indicators in order to use indicators correctly. In this section, indicators are discussed one at a time. You will learn when to use indicators, how to specify them, and which program cycle operations are associated with each type of indicator.

TRANSACTION					
DATE	LIEM	DESCRIPTION	QUANTITY	UNIT	SALES
DATE	NU			PRICE	AMOUNT
7/23/	413010	CHOD1 BOX 1004 FLUSH	10	4.90	49.00
	412146	CH148 BREAKER 15A	100	.89	89.00
	411116	1500 TWIN SUCKET B	500	1.12	560,00
					698.00
7/24/	503029	MUTUR 1/2 HP 60 CYC	2	146.78	293.56
	327802	TERMINAL CLIP	100	5.12	512.00
	326013	TERMINAL BAR	100	4.12	412.00
	411121	1506 SOCKE ADAPT BRN	400	•19	76.00
	412997	CH173 BREAKER 3UA	60	1.15	69.00
	413088	CH176 BREAKER 60A	40	1.15	46.00
	411174	C151 SIL SWITCH BRN	200	1.16	232.00
	413090	CHOO5 BR BOX 150A	10	4.98	49.80
	718326	FC803 FUSE 15A	200	• 3 2	64.00
					1+754-36
7/27/	321071	2-SPEED SAW	1	28.44	28.44
	325781	SATIN-CUT DADU SET	1	39.50	39.50
	412146	CH143 BREAKER 15A	50	•89	44.50
	573022	6-VOLT POWER BATTERY	2	14.45	28.90
					141.34
				A 1	2-593 70

Note: This report is similar to those shown before, but note the addition of headings and totals.

Figure 11. Printed Report

CONTROL LEVEL INDICATORS

Control level indicators are used when you want to calculate and print totals. Nine different indicators can be used (L1 through L9), allowing as many as nine different levels of totals in the same program. The control level indicators tell the program two things:

- · When totals should be calculated.
- · Which calculations and output operations are total operations.

A control level indicator in positions 59 and 60 next to an input field specified on the Input Specifications form determines when totals should be calculated and printed. This input field is called a *control field*. Whenever the contents of the control field changes, a *control break* occurs. A control break turns on the control level indicator assigned to the control field and all lower control level indicators. The system performs all calculations and output operations (total operations) that are conditioned by these control level indicators. If a control break causes L3 to be turned on, L1 and L2 are turned on. This allows control over several levels of totals and subtotals. Examples are daily, weekly, and monthly totals.

In Figure 11, the records are grouped by date and a total of the sales amount field should be printed when the contents of the input record date field changes.



Control level indicators are specified at (1) to tell the compiler when total operations are to be done. They are used at (2) to tell which operations are total calculations. They are used with a T in position 15 at (3) to tell which lines are total lines.

Program Cycle Operations

Figure 12 shows the program cycle operations associated with control level indicators. The system can do calculations and output operations at two different times in one cycle: at detail time and at total time. Total operations associated with control level indicators are not done in every cycle; they are done during the cycle in which the control field changes.

After a record is read, the program determines whether the control field in the record just read is different than the control field in the previous record. If it is, a control break occurs and the control level indicator you specified is set on. When the indicator is on, it means that all records in the control group have been read and total operations can be performed. Control level indicators remain on through the detail calculations and output processing of the record that caused the control break. They are then set off before the next record is read.



Figure 12. Program Cycle Operations for the Control Level Indicators

Detail operations for the record that caused the control break are done only after total operations for previous records. The control level indicator assigned to the field that caused the control break remains on so that the first record of the group may be identified with that indicator. The record that caused the control break is not processed before the total operations are done, because information from that record would be included with information from records in the previous group. The totals from the previous group would then be wrong.

RPG II Specifications

To specify a field as a control field, you assign a control level indicator (L1-L9) to an input field in positions 59-60 on the Input Specifications form:

GX21.9094 UM-0501

NPUT SPECIFICATIONS



L1, assigned on the same specification line as the date field, tells the system to use DATE as the control field.

To specify which operations are total operations, you assign the same control level indicator in positions 7-8 on the Calculation Specifications form and in positions 24-25, 27-28, or 30-31 on the Output Specifications form. On the Output Specifications form, you must also enter a T in position 15.



You can specify up to three different indicators on a line on the Output Specifications form. If you are using only one indicator, you can enter it in any one of the three positions. The control level indicators specified on this form can be used to condition an entire output record or only certain fields in the output record.

Keyan Keyang Graphic Card Electro Number Programmer Date Instruction Key	Page d Program dentification
Image: Second second	Harmons No Sup CR - X Hermove 1 A J Y Date Plus Sup 2 B K Fred bit Fred bit 3 C I, Z Auron 3 C I, Z Auron 3 C I, Z Auron Fred bit 3 C I, Z Auron Fred bit 3 C I, Z Auron Fred bit 4 D D D D D D 4 D D D D D D 4 D D D D D D 4 D D D D D D D 4 D D D D D D D 4 D D D D D D D 4 D D D D D D

Using the Blank-After Specification

In RPG II, you can set fields in storage to blanks (in the case of alphameric fields) or zeros (in the case of numeric fields) after they have been written out. You do this by entering a B in position 39 of the Output Specifications form.

This is a particularly useful feature when you are doing total operations. It allows you to use the same field over and over for accumulating and printing totals. For example, you could use a numeric field to accumulate totals for a particular group of records. After the totals are accumulated and printed for that group, you can use the same numeric field to accumulate the totals for the next group of records. To do this, place a B in position 39 for the total field. If you do not place a B in position 39, the totals for the second group of records.

JOB 3 (TRNREG): USING CONTROL LEVEL INDICATORS TO CALCULATE AND PRINT TOTALS

Job Definition

Print a weekly sales transaction report that lists all daily transactions and gives the total sales for each day. This report is similar to the reports produced in Jobs 1 and 2. All items sold each day are listed. Item number, item description, quantity sold, unit price, and sales amount (quantity times unit price) are included for each item. The date is printed only for the first transaction encountered that has a new date. The total sales amount for a day is printed after all transactions for that day have been recorded.

Job Requirements

Input: Sales transaction file consisting of 96-character records. The records are arranged in ascending order by date. The format of the input records is shown on the record layout form:



Processing:

- · Multiply quantity times unit price to find sales amount.
- · Find total of all item sales per day.

Output: A printed transaction register:



This printer spacing chart shows how the report is formatted:



Job Specifications



RPG CALCULATION SPECIFICATIONS



FIRST PAGE INDICATOR

The first page (1P) indicator is used on the Output Specifications form to specify the headings you want printed only on the first page of a report. Headings, usually printed at the top of the page, include such things as report titles or column names:

I'RM	RPG OUTPUT SPECIFICATIONS	GX21 9090 UM/050 Printed in U S.A.
Program Progra	Keying Graphic Card Lieuro Number 1 2 Instruction Key Final Card Lieuro Number Page Page Final Card Lieuro Number	75 76 77 78 79 80
O I <thi< th=""> <thi< th=""> <thi< th=""> <thi< th=""></thi<></thi<></thi<></thi<>	Output Indicators Field Name And And And And And And And And Sign Commax And And And And Sign Commax And Commax And Commax And Commax Sign Sign Sign Commax Sign Commax Sign Sign Sign Sign <th>21 24 DRI NJ 70 21 72 73 24</th>	21 24 DRI NJ 70 21 72 73 24
	ft at the top of the paper in order to avoid printing close to the perfora-	
1 6 0 1 7 0 1 8 0 1 9 0 2 0 0	inserted into the printer so that the first print line will be just below the kip to line 06 before printing in positions 19 and 20 will begin the print- the perforation (assuming six lines to the inch).	
Program Cycle Operations

One operation in the program cycle is concerned with the 1P indicator (see Figure 13). The 1P indicator is automatically set on at the beginning of every job, so the first operation taken by the system is to print any output record conditioned by 1P. After this is done, the first record is read and the program cycle operations are executed in order.

Headings conditioned by 1P are printed only once-at the beginning of the job on the first page of the report. Any heading records that are not conditioned by 1P are handled in the same way as detail records. This means that they will be printed along with detail records in every cycle.



Note: The first operation in the first program cycle concerns output operations conditioned by the first page (1P) indicator.

Figure 13. Program Cycle Operations for 1P Indicators

RPG II Specifications

Heading information to be printed on the first page of a report is specified by using constants (actual information instead of field names). Constants for headings must be specified according to these rules:

- Constants must be entered in positions 45-70 of the Output Specifications form.
- Constants can contain any of the characters in the data character set (the 256 EBCDIC characters).
- Constants must be enclosed in apostrophes. (The beginning apostrophe is always entered in position 45.)
- No field name can be used on the same line as a constant.
- · An end-position entry must be entered for every constant.

	Field Name	End Positio	n	Commas Zero Balances to Print No Sign CR - X Bein Sign Yes Ym 1 A 3 Y Eate Yes No 2 B K Function No 2 B K Function No No 2 C L Function No No 4 M Scatores Scatores		
-	*aUTO 12 33 34 35 36 37	3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	t 1/9/9/4 43	Constant or Edit Word	71 72 73 74	
		1	5	CUSTOMER NUMBER -		Valid constant.
		3	5	WAME /		Invalid constant. The constant must be preceded by an apostrophe in position 45.
	COST	4	5			Invalid constant. A field name cannot be specified on the same line as a constant.
		a la	ø	∑2'		must be followed by an apostrophe. Valid constant.
		4	3	'DAILYTRANSACTIONREGISTER'		Valid constant. Although the constant is specified correctly, it will be unreadable when printed
						because no spaces were left between words.

Headings too long to specify on one line of the Output Specifications form can be split and placed on separate lines. You must, however, give an end position for each part.

The heading shown in the printer spacing chart takes 28 positions. A constant containing a maximum of 24 characters can be specified on one specification line. Because the entire heading cannot be specified on one line, it must be broken into parts. The examples given show three different ways to specify this heading:



Heading lines should be specified first on the Output Specifications form. The best method is to specify your record types in this order: heading, detail, and total.

OVERFLOW INDICATORS

You use overflow indicators to:

- Print headings on every page but the first page of a report (the 1P indicator allows headings to be printed on the first page).
- · Control where printing begins and ends on a page.
- Advance forms from one page to the next (provided a skip specification is also used).

To understand how overflow indicators work, you must know how the concept of overflow is defined in RPG II:

- · Overflow-Lines that remain to be printed after a page is full.
- Overflow handling-Advancing forms to a new page after the last line has been printed on the current page.
- Overflow line-The last line to be printed on a page.
- · Overflow page-The new page to be printed when overflow occurs.



Printers use continuous forms (a series of pages divided by perforations). Overflow handling refers to the means of advancing forms from one page to the next.

Overflow can be handled automatically by the system or through specifications you write.

Note: This discussion assumes that you are using standard 66-line forms in the printer.

Printing always begins on line 06 (assuming the operator positions the first page properly) and ends with line 60. Overflow occurs after line 60 is printed; that is, forms advance to line 06 of a new page.

When overflow is handled automatically, heading lines can be printed on the first page if 1P is used. No instructions are provided, however, to print headings on overflow pages. Heading lines are considered to be detail lines and will print on each detail cycle if the conditions placed on printing are met.

When you do not want overflow handled automatically, you can specify on the coding forms how you want it handled, using overflow indicators:



File Description Specifications

Assign an overflow indicator to the printer at (1), then use it on the Output Specifications form at (2) to show which operations must be done when overflow occurs.

Program Cycle Operations

Figure 14 shows operations in a program cycle in which overflow indicators are used.

The program sets on the overflow indicator you assigned whenever the overflow line is passed. By setting the overflow indicator on, the program remembers that overflow has occurred. As you can see in Figure 14, overflow indicators can be set on at one of two times: at detail time when a detail record prints on the overflow line or at total time when a total record prints on the overflow line. Notice that the only time a check is made to see if the overflow indicator is on is right after total output. If the overflow indicator is on, overflow operations are done in this order:

- 1. Print any total lines conditioned by the overflow indicator.
- 2. Skip to new page, provided a skip specification was made on a line conditioned by the overflow indicator.
- 3. Print all heading and detail lines conditioned by the overflow indicator.



Figure 14. Program Cycle Operations for Overflow Indicators

RPG II Specifications

There are eight overflow indicators: OA through OG and OV. You can enter any one of these indicators in positions 33-34 on the File Description Specifications form. If you have more than one printer file, however, you must specify a different overflow indicator for each file.

After you have specified an overflow indicator on the File Description Specifications form, you must specify the same indicator on the Output Specifications form. This specifies what you want done when overflow occurs.

Besides specifying the overflow indicator, you must also specify that forms should advance. You do this by placing a skip specification in positions 19-20 on the Output Specifications form.



The skip specification should be made on the first line you want printed on the page (usually a heading line). The skip entry is the line number of the beginning line (usually 6).

Always remember to enter a skip specification for advancing forms in a heading line conditioned by the overflow indicator. If you forget, forms will not advance when overflow occurs.

Using Spacing with Overflow

You already know that the overflow indicator is turned on when a record is printed on the overflow line. However, this indicator also turns on whenever the overflow line passes under the printing mechanism. This means that spacing to a line past the overflow line to a line on the same page causes the overflow indicator to turn on. Figure 15 shows an example of spacing after printing.

IBM Interstated Respect Mediums (and		RPG	OUTPUT	SPECIFICATIONS	GX21 9090 - UM-050* Printed is U 5 A
Program	Date	Keying Instruction	Graphic Kry	Cand Dectro Number 1 2 Page [[]] μage [[]]	75-76-77-78-79-80 Prospann [
O f dename 1 0<	me, for exan detail line spectrum di wo spaces. S flow line (line	Output Indicators	Field Name	tomme Zeneration Machine Machi	X Be move Y Lot Z Zero 3 "opported" Be 10 Zero Be 10 Zero

Figure 15. Spacing after Printing

Using Overflow and 1P Indicators Together

The overflow indicator is most often used with other indicators. However, we will discuss the use of overflow indicators only with the 1P indicator. Both 1P and OV cannot be on at the same time.

If you want headings on all pages of a report, you can use both the 1P indicator and an overflow indicator. 1P causes headings to print on the first page; the overflow indicator causes them to print on all succeeding pages. If a record should be printed when either one or another condition occurs (either 1P or OV is on), you can specify indicators in an OR relationship.

IBM	International Bosin	est Marborne Course			RPG	OUTPUT	SPECIFICATIONS (ax.	21.9090 - 3M-050* and in U.S.A
Program Program	n Inner		Date		Keying Instruction	Graphu Key	$\left[\begin{array}{c c c c c c c c c c c c c c c c c c c $	76 77 78 79 90
C Line	A A A A BO	ame	z = 0 $> C = z$ Sikr= Feter Fr z = 0 m Beton $0z = 0$ $Arter = 0$	Skip aug 30 30 30 31 32 21 22 21 22 32 32 32 32 32 34 34 34 34 34 34 34 34 34 34	Output Indicator	• Field Name	Communication Communication Non-Supple C.B. A. Permove Vis. Vis. <td< th=""><th>4 - 1 1 - 10 - 72 - 72 - 74</th></td<>	4 - 1 1 - 10 - 72 - 72 - 74
0 1 0 2 0 3 0 4 0 5 0 5 0 6 0 7 0 8 0 8			3		1P 0V 1P 0V		25 SALES REPORT' 4 'ITEM' 20 OTY SOID' 36 'TOTAL AMOUNT'	
1 0 1 1 1 2 1 3 1 4 1 5 1 6 1 7 1 8 1 9 2 0		If a rec indicat case) o (OV in fields a will pri Space a OR lin	cord c or in the this c nd co nt wh and sk e also	an be an OR same case) o onstant nen eit kip ent apply	written when relationship line as the lin n the followi is to be includ her 1P or OV ries are not r to the OR lin	n either one of . Specify the in he type. Place the ing line with the ded in the reco / is on. hecessary in the ne.	f two conditions exists, place the conditioning indicator signaling one condition (1P in this the indicator signaling the other condition he word OR in positions 14-15. Then specify ord in the normal way. In this example, headings he OR line. The entries in the line preceding the	

LAST RECORD (LR) INDICATOR

The last record (LR) indicator is associated with end-of-job procedures. The program uses LR to indicate that the last data record has been read and that end-of-job processing is to take place.

Use of the last record (LR) indicator is optional. When LR is not used in specifications, the compiler automatically supplies end-of-job instructions. If you use LR, you are indicating that certain operations, such as printing a total count of all records read, must be done after all input records are processed.

Program Cycle Operations

Figure 16 shows the operations in the program cycle associated with the last record indicator. RPG II is set up so that it senses an end-of-file record containing some identifying information to indicate end of the data file. Each data file has an end-of-file record.

Whenever a record is read, the program checks to see if all records in a file have been processed. If all records in all of the files have been processed, the program sets on all control level indicators L1-L9. It also sets on the LR indicator to indicate that all records have been processed. All total operations (those conditioned by LR and L1-L9) are performed. After total operations have been done, the program checks to see if LR is on. If it is, processing stops.



Figure 16. Program Cycle Operations for Last Record (LR) Indicator

RPG II Specifications

The LR indicator is specified by an LR on the Calculation Specifications form or Output Specifications form. This entry specifies which operations are to be done after the last record is processed:



The LR indicator is specified at (1) to tell the system which calculations are to be done after the last record is processed. The LR indicator is specified at (2) to tell the system which output operations are to be done after the last record is processed.

JOB 4 (TRNREG): USING FIRST PAGE, OVERFLOW, AND LAST RECORD INDICATORS TO PRINT HEADINGS AND TOTALS

Job Definition

Print a weekly sales transaction report that lists daily transactions, total sales for the day, and total sales for the week. This report is similar to the one created in Job 3. The only difference is the addition of headings and final total.

The report title and column headings are printed on every page of the report. All items sold each day are listed. Item number, item description, quantity sold, unit price, and sales amount are included for every item. The date is printed for the first transaction in each group. After all transactions for a day are listed, the daily sales amount is printed. A final total of all daily sales is printed at the end of the report.

Job Requirements

Input: Sales transaction file consisting of 96-character records. Records are arranged in ascending order by date. The format of the input records is shown on this record layout form:



Two decimal positions

Processing:

- Multiply quantity times unit price to find sales amount.
- · Accumulate sales amount to find total item sales per day.
- Accumulate total daily sales to find total weekly sales.

Output: A printed transaction register:

		TRANSACTION REGISTER			
TRANSACTION DATE	ITEM NO	DESCRIPTION	QUANTITY	JNIT PRICE	SALES AMDUNT
7/23/	413010 412146 411116	CHOO1 BOX 100A FLUSH CH148 BREAKER 15A 1500 THIN SUCKET B	10 100	4•90 •89	49.00 89.00
	411110	1300 TWIN SOCKET B	500	1.12	698.00
7/24/	503029 327802 326013	MUTUR 1/2 HP 60 CYC TERMINAL CLIP TERMINAL BAR	2 100 100	146•78 5•12	293.56 512.00
1/21/	321071 325781 412146 573022	2-SPEED SAW SATIN-CUT DADD SET CH143 BREAKER 15A 6-VOLT POWER BATTERY	1 1 50 2	28.44 39.50 .89 14.45	28.44 39.50 44.50 28.90
					141.34
			WEEKLY TOT	AL	2,593.70

This printer spacing chart shows how the report is formatted:

	,	2345676	1 1 1 1 1 1 1 1 1 1 1 9 0 1 2 3 4 5 6 7 8 9	2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 0 1 2 3 4 5 6 7 8 9 0 1 2 3	3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	555555555566	66666667777	77778	
1	П		┥┥┫┥┥┥┥┥┥┥┊┊	╈╋╗╗					
2	П		╡╡╏┊┇╡┇╡	╺╋┾┾┼┾┾┾┼┼┼╂╂┼┽	╶╏╞╏╞╞╞╋╋┥╞╞╪╞ ╪╪	╊╆┼┾┾┼┼┾┾╋┥	╍╋┽┼┼┼┼┼┼┼╉╂┼┼┥	╺╋╪╪┽╪╋	┥╡┊┊┊┊┊╡╡┋ ╋╡╋
1	Π		, , , , , , , , , , , , , , , , , , , 	╶╋╡┼┼┼┼┼┼┼╋┧┼┼	╶┊┊┋┊┊┇┋┋ ┇ <u>┊</u> ┊┊┊┊┊┊	╊╁┼┼┼┼┼┼┥	╶╋╊┼┿┽╋╫┿╋╋┼┼┙	·┼┼┼┼┼╂	┼┼┼┼┼┼┼┟┟┟╋┾┿
4	Π				╶┼┼┦┦┦┦┩╵╎┶┼┦┼┊┝┤	╉╪┾┼┼┾┼┾┽┽╋┥	╺╊╄╄╄╄╋╋╋	╺╋╗╡╋┿┿╋	┼┽┽┽┼┼┼┼╊┦┼
-5							╺╋┟┽┼┥┥┥┥╻	╶┼┊┽┼┽┽╉	<u>┥┥┥┥┥┥┥┥</u> ┥┥
	Ц				TRANSACTION	REGISTER		╶┼┼┼┼┼╉	<u>┥┥┝┽┼┽┽┽</u> ┫┤┼
7	Ц	++++++							
<u></u>	Ц	╺╅╉╅┟╽╽╻	TRANSACTIO	М	DESCRIPT	ION	QUANTITY	UMIT	SALES
<u></u>	H	╺╅╂╂╋┠┼	PATE	Mp				PRICE	AMOUNT
	+	┝┾╇┽╇┿┿	╅┽╉┼┼┼┼┼┼ ┼	╶╉┼┼┊┿┿┼┾┿┿┨┿┽┽			╺╁┾┽┼┆┠╢┟┫┥┼╵	╴┧┽╿╌┽┾┿╋	┿┿┿┿┿┼┽┼┥┠┿┿╌
- 7 10	Н	╺┾┽┼┾┿┿	++ WAAAAA		+++++++++++++++++++++++++++++++++++++++				
-	\mathbf{H}	╺╋╋╇╋╋┿				XXXXXXXXXXXX	<u> XXXXX</u>	XX · XX	
	+	┝╋╋┼╊╋┼	+++ ADATED ++	1 17 TEAN WILLAST	ot the plan		Tour ut the	toptal	
119	H	┝╋╋┾╋╋╋	++++****		M - 1 (GM /2634K)			ALL WALL	AMOUNT
7 10	H	┝╋╋╋╋╋	╅┼╋┼╃┽┽┽┼┼┼	╶╉╄┾┽┽┽┽┼┼┼┾╊┽┿┼	╶┨┧╏┿┿╊╋┝┽┍╄┩┥┥┥ ╡	╋┾┿┾┿┿┿╋┩	╺┾┿┼┼┽┽┟┽┲╂┼┽┙	*****	
17	П		┽┽┲┼┽┽┼┽ ┊┤┊┤	╶╊┞╪┽┼┼┼┿┼┼╕┼┼┼	╺╋╞╄┝┿╞╊┊┍╪┊┊ ╞┾	╺╋╄╋┿┲╞┽┼┼┽╋┩	╺┿┿┿┿┿┿╋╇╋┿┿	·+++++₽	
1					┼┼┼┼┼╀╊ ┟┼┼╎┼╎┼		╶┼┼┼┼┼┼┼╋┼ ┼	1024	VLVSALES
19	IJ								
_T 20	Ц						WEEKLY TOTA	4 X.D	
21	Ц	┝╇┿┿╇╇			<u> </u>				
22	4	┝┿┿┾┾╄┼	III PINAle	space det	ail lines.	╏╷╷╷╷╷╷╷╷╷	╶┽┼┽┽┽┽┽╋┽┽╴	LWEE!	KUY SALESA
	4	┝┼╂┾╂┼┼	┽┽╂┼╈┥┼┤╻╻╸	╶╉┶┥╁╁╎┙┶┥┝┛┥┢┥	╶╪╧╘╤╡┽╏┾╱┼┥┅┼┾┼┼	╻╡┊┽<u></u>╡╻┊╻┊	╺┾╼╍┊┊┊╸	┈┼╺╴╎┥┝┿╉	┼┿┼┿┽┿┥┾╸
	-	┝╋╋╈┿╋╇	Vauble	space peru	leen detail and	1 total line	2	┥┥┥┥┥	┿┿┿┿┿┿┿╋┿┿
	+	┝╄╄╼╁┽┽	┼┼╊┼╧╡╧┼┾┶┢╸┤		┲╾╾╼╼╌╴╴╴	┨┧╽┼┾┝┼┼┤身┤		• ┿┿╄┿╄┿╋	┊┊┊┊┊┊┊┊┊┊┊
		┝┿╋┽┽┽┽	++++4/74/84	IZNAGE DET	rare printing	HINGI WAG	KIY TATAL	╺┽┿┿┾┽┾╉	┽╁┊┽┽┼╞╞┾╸╉┿╼
- 21	H	┝┿╋┼┿╇┿	╡┤╊┤┼┼┼┽┥ ┥	╺╉┼┾┿┿┽┽┼┼┿╋┿┽┼┾	╶┥┝┼┢┝╞ ╉┾ ┍┾┥ ┊┊┊	╉┿┿┽┿┿┾┾┿┽╋┥	-+FĨ++++ + +++	╺┾┑┾┽┽┾╂	┼┼┼┼┼┼┼┼╉╇┾╴
	H	┝╞╋┊╡ ╋	╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋╋	╺╋┿┼┼┾┼┼┼┼┿╋╄┼┥	╶┼┼┼┼┼┊╏╞╏┊╎┊ ┊┊┥	┨┼┾┽┾┿┼┾┾┿╋┥	╺┽┿╊┿┿╋╋╋┿┥	· ┼┊┤╿╿╿	┥╿╿╗┇╎╎╎╎╏╹┥

JOB SPECIFICATIONS





RECORD IDENTIFYING INDICATORS

In the jobs discussed so far, we have assumed that all records in the input file were alike. They did not necessarily contain the same information, but they had the same fields and the same kind of information in each field. They were of the same *type*.

However, in real jobs, often data files will not contain records of all the same type. Files often contain many types of records with different fields and different information. When using different record types in a job, you must have a way of telling the system what operations (calculations and output) you want done for each record read. Record identifying indicators are used for this.

Program Cycle Operations

Figure 17 shows program cycle operations associated with record identifying indicators. A record identifying indicator is set on right after a record is read and is set off before the next record is read.

Normally, record identification indicators condition detail calculations and detail output operations because detail operations are done for the record just read (the one associated with the record identifying indicator). On the other hand, total operations are not performed for any one record type; they are done after a certain number of records are processed.



Figure 17. Program Cycle Operations for Record Identifying Indicators

RPG II Specifications

These are the RPG II specifications you must make when using different record types and record identifying indicators:



Assign a record identifying indicator (a 2-digit number from 01-99) to each record as (1), then use that same indicator to condition calculations at (2) and detail output operations at (3), which must be done for that record only. Identify each record type by describing a code which identifies it at (4). If records must be in a certain order, indicate that sequence at (5).

As we said earlier, you must describe each type of record in your input file. This description includes the names of all fields used from the record, the location of the fields, and the type of data in the fields:



When you describe fields from each record type in the input file, give a unique field name to every *different* field from all record types. If, however, two or more record types contain identical fields, you can assign the same name to the field in each record type. Only one storage area would be assigned for the same field from several record types, but it would not matter because the information in the field is the same.

You must do more, however, than merely list the fields in all records because the RPG II compiler would not know which fields were in each type of record. Furthermore, it would not know which record it was reading. To give the compiler a means of identifying records, you use record identification codes and record identifying indicators.

Specifying Record Identification Codes

When you create records, you should include an identification code in each one. For example, to identify an item transaction record, you might place the code TR somewhere in that record. You can use any combination of letters and numbers for the identification code and you can place the code in any record positions.

When you describe the record on the Input Specifications form, you use positions 21-41 to describe the record's identification code and the location where the code is located in the record:



For each character in the code you must specify:

- (1) Where in the record the character is found.
- (2) What the character is.
- (3) The letter C to indicate character.
- If the code requires that a character must not be present, enter N before the letter C.

You can specify a code of up to three characters on one line. If your code contains more than three characters, use the next line and the word AND in positions 14-16. Figure 18 shows some examples of how to specify record identification codes.



Figure 18. Valid Specifications for Record Identification Codes

Specifying Record Identifying Indicators

You specify a record identifying indicator in positions 19-20 for each record type used in the job. Record identifying indicators are numbered 01-99. Use a different number for each record type.

A record identifying indicator is specified on the same line as the identification code. All fields for the record are then listed, starting one line below the identification code and identifying indicator. The file name need be specified only once, on the specification line describing the first record in the file.



After reading a record, the program checks the identification codes to determine which record it has read. When it finds a match between the code on the record and the code stored from the Input Specifications form, it turns on the record identifying indicator associated with that record. This is the program's way of remembering which record it read.

Record identifying indicators can be used to condition calculations, output records, or output fields. In this way, the program performs the proper operations for each different record type.

When you use record identifying indicators to condition calculation and output operations, you are assured that these operations are done only for appropriate records. If you do not use these indicators, operations will be done on all records.

Record identifying indicators should be assigned even when there is only one record type in a file. If you do not use them, the compiler prints a message telling you that record identifying indicators have not been assigned.

The record identifying indicator should also be used to condition detail output records. This prevents detail lines from being written on the first cycle. If the detail line is not conditioned by a record identifying indicator, any constants you specified on the detail lines would be printed even though the first record has not been read.

Specifying Record Type Sequence

Sometimes records must be in a particular order within a record-type group, but at other times the order makes no difference. When records need not be in a particular order, enter two alphabetic characters in positions 15-16. Different alphabetic characters can be used for each record:

Γ	1	Γ		Ι										T			Γ	ſ		5	Τ							R	ecc	ore	4 H	de	ntil	ica	itic	'n	Co	bd	es					-		Ţ											П			
┝	-	-	٦																						1								2				Τ				2	,							۲	16	d	La	юа	tio	'n					
	Lir	ne	Form Tucu	arth Linne -			Fil	ler	าลเ	m	e		G	>	Secure		Number [1/N]	Ontion (O) 11 S	Parcel Ideat	SU V	5	R	, Iier	lo	•	Not (N)		G(2/D	Character		P.;	<u>ii</u> tiu	m	No. (N)	CDB	Character	Cital dLIE	P	o ¢İ	1100	ħ	Not (N)	C710		Character	NAMES STRATC	P/8/L/R	-	Fre	 6.0	n.			To	D		Decimel Positions	F	ie	ld I
3	4	5	6		, ,	3	9	10	11		12	13	14		5		17	18	19	20		1 3	•	23	24	25			27				• •	3:	3	3	4	15		37		39				2	43	44	45			7 4	18	49	50	51		63	54	54
0	1		1	Ĺ	N	1	0	Ü	7	ŗ		_	I		9	9			1	0	\$	T	1		1		Ĩ		A		T	Т	Τ	Î	Г	T	T	T	····	Γ	Г	T	Ĩ	Ĩ					Ċ	Г	T	t	T	Ť	T		Ť	Ţ	Ī	Ť
0	2		1	1	Τ	1	1		T	1	1		t	Î	T			ſ	Γ	Ť	ľ	T	T			ľ		Ĩ			T	T	T	t	T	1	ľ				t	t	t		- Andreas		-		Ē	t	t	Ť	-†	+	-+	-1	1		+	+
0	3		1		Ī	Ţ			ľ	Ţ			†		T			-	t	1	t	t	1			t.	t	1			t	t	t			t	t	1		Ē	t	ŀ	t						m	t	t	1	+	· - +-	 i	-			+	+
0	4	Τ	I	:	1	Ť	1		Ť	T	1	_	ľ	Ì		3	•		2	0	¢	Ť	1	-	Ï	t	Ì		R		t	t	t		t	t	t	1	-	1	t	t	t		ľ	1			m	ł	+	đ	-+-	+	-+	f	1	• •	• +	-+
0	5	+	I	-	+	1	1		 	ļ	-+		+- : :		1		_		F	Ĺ	ţ,	1	1			† -	f L				Ļ	Ļ	ţ		t				1		İ.				-					L	t	ł	+	-+-	-+				+	+

If the records must be in a specific sequence, you must make entries in positions 15-18.



- Positions 15-16 (Sequence) contain numeric entries (01-99).
 The sequence entries must be in numeric order.
- Position 17 (Number) tells how many records of this type to expect in the record-type group:
 - 1 = One and only one N = One or more
- Position 18 (Option) indicates whether the record type is required in the record-type group:

Blank = Record type is required O = Record type is optional

Positions 15-16 should have an entry for every record type specified. If there is only one type per file the entry should be alphameric. If you fail to use a sequence entry, the compiler prints a message saying that no entry was made in positions 15-16.

JOB 5 (STOKST): USING RECORD IDENTIFYING INDICATORS TO PROCESS DIFFERENT RECORD TYPES

Job Definition

Print a stock status report. This report is printed whenever the inventory is updated. It gives detailed information on all active merchandise. The first line for each item in the report shows standard descriptive data for the item: item number, item description, quantity on hand, and quantity on order. This information is taken directly from the input record.

Subsequent lines give the detail on current transactions involving the item: sales to customers and receipts from suppliers. This information is also taken directly from input records.

Quantities remaining on hand and on order are calculated for each item and printed after all transactions for the item are listed.

Job Requirements

Input: An inventory file consisting of three different types of records. Formats of the three record types are:

Name ITEM MASTER RECORD





Name RECEIPT RECORD

CODE - R	ltem Number	Quantity Received	
	2 3 4 5 6 7	8 9 10 11 12	13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 10 41 42 43 44 45 46

The file is organized in ascending order by item number. For each item, one master record is required. Issue and receipt records are optional. When present, however, there may be any number of each. Records for each item are in this order:

- 1. Item master
- 2. Issue(s)
- 3. Receipt(s)

Processing:

- Find total number of each item sold. To do this, perform the calculation ISSUE + TOTAL ISSUE = TOTAL ISSUE for each issue record.
- Find total number of each item received. Perform the calculation RECEIPT + TOTAL RECEIPT = TOTAL RECEIPT for each receipt record.
- When all transaction records for one item have been read, find new quantity onhand (ONHAND + TOTAL RECEIPT - TOTAL ISSUE = NEW ONHAND) and new quantity on order (ON ORDER - TOTAL RECEIPT = NEW ON ORDER).

Output: A printed stock status report:

LTEM NO	DESCRIPTION	QUANTITY ON HAND	QUANTITY ON ORDER	TRANSACTION QUANTITY
+11116	B500 TWIN SOCKET BLUE ISSUE RECEIPT	458	500	50 500
		908÷÷	0**	
11122	8506 SOCKET ADAPT BRN ISSUE ISSUE ISSUE	325	o	20 38 10
		257**	0**	
11173	CI5IC SIL SWITCH IVORY RECEIPT	50	150	150
		200÷¢)**	
11254	AZIO PULL CORD GOLD ISSUE ISSUE	62	75	16 30
		16**	75≎¢	

This printer spacing chart shows the format of the report:

		2	3 4 9	5 6	78	9 0	1 1	2 3	1	5 6	1 7	1 1	2	2	2 2 3	24	2 2 5 6	2	2 2 8 9	23	3	3323	34	3	36	33	3 9	4	4 4	43	4	4456	47	4 8	4 5	5	52	55	55	5 5	5	5 6	56	6	5 6 3 4	6 5	667	6	6 7 9 0	7	7 2	7 7	7 7 5 6	7	78
	1	T						+		1	T	1	1						1	1	Π		T	Π	1						Π					Γ				ļ	-		Τ												
	2						+	1		1	††	1	1	T	+	1		+	1	+-		1	T	11		1			1			Ţ	Ţ			Γ		1		T	Π		Τ	Π	Ţ	T	Ţ						,		ĺ
	3			T							T				1		1		-+-	1					1								1			Γ							Γ							Γ				T	, i
	4	T						1		1	••••				T		T			1		T	1	Π		1							Τ			Τ		-					Τ	П	i					Ι					
	5				+ +	11	•	1		1		•	1			tt	-		i	- +	Π	-	T			1				1	Ţ	1	1	I			1			1			T	I	T			Τ	Ī	I					
H	6			ŤŤ				1	+ † 1 1	-	11		- 4			1	1					1	1				S	Т	00	K		S	T	A	TL	١S		F	{ E	PC	R	T	Ī		I			Ī	!	Ι					
	7				11			1		1							1		Ţ		Π	T	Ι	U			1			1		T	Ì			Ι		Ţ							-							_		-	Ì
				11												1			Ì	1			I							Ţ	Π											 	1.		1			4		1	ļ	_			Ļ
H	9		IT	EM	N	0		ĺ	Li		ÌÌ	DE	S	C	RI	Ρ	T'I	0	N	Ì.]			QU	A	N	T']	ĽΤ	Y				Ql	IA	N	T 1	T	Y			Ţ	RI	ΗŅ	5	A C	. T	11	ŊΝ	1,	+		• •	+ · +
<u>H</u> 1	0			-				÷											-	-						0 N		Η	AI	۱D)		_	H	01	4	0	RI)E	R	1	: +-+	<u></u>	u/	٩Ņ	ļΤ,	17	Y	! † +		ļ.,			+ +	Ļ Ļ
	11		44	4				-	L	-+	-				+	Ļ	-	+		+		_		 +-+		_			1	+		+					! • •				.+	∔ -+	4.	: +-+		.	+		: • · •			· +		· · · ·	• 4
D	12		XХ	XX	XX	-		X	X	XX	X	X	XX	Χ	χ_{\downarrow}	X	X٢	<u> </u>	X)	۲X	X	X)	X	Ļ	4	X	X	,	X.)	(X	4		+	+	: ! 	X	X	<u>, x</u>	X	<u>X</u> ,	-	••	1	: + +				+	 .	+	+-+				⊷ +
D	13	-	$\left \right $	\downarrow		- -		ļI	5,	ςι	ĮΕ			Ļ		Ļ		-+		+-		_				_		-		_+_	+-+	 +	-	+ +	+	1.		+-	+	↓ ∔	÷	••		÷ +	Xγ	(,)	X/	X	+ +		÷.,	+		; .+ -	••
D	14	_	i - i - i				L.	R	E	C¦E	ļ	PŢ	r			+	-		i i	+		1	_	+-+	+ +	+	-	Ļ			+		-	+ -		1	++		_	+ - +	+	•	4	÷	X	.,	Χ,/	XX	, +	4.	-+ -+	: +-	• •	• •	+ +
	15		44	\rightarrow	-				44		-	i 	: -+-	H		+	+		-	1		_	+		Ļ					-	7-2-1	ا + یز	- -	++	;		÷.,	;		\/.		` †→	+	; ; ;	- ÷	÷		-		+	+ +	+	• •	+	+ +-
T	16	-	↓ ↓ .,	-+-+				-+	+ +					1.	÷.	+	Ļ.		:		L	-+	+-	1		λ	(X	,	X,	X;)	(*	π ⊢+-	-		1	<u> </u>	X	• ; ;	XΪ	X_{+}	- +	ļ.,	-	+-+	÷	+ -•	•• ••	+	ίt		+-+		• •	÷	+ - +
	17	+	i i i	-++	• - • •	-+			Ļļ	•	: 		-+	+ +	+	-	 	+	ŀ+	-+	+		-		4	NE	W				<u></u>	i	- i	÷	1 - 1 † -†	IN	E	W		+	rto	5	-	+ -+			••••	÷	+ +	-+	+-+	+	+ -+	•	• • •
	18	+	∔∔-•	-+-+	-+		┞	+	+		+-		+	\downarrow	+	÷	Ļ	÷	H	+	+	+	+-	-	Ļļ	0	4	Щ	н	V L	2		•	+	∔ -+	4	^N	+	2 <u>'R</u>	+++	E K	4	+	+ +	-+-		•+	+-	÷	╋	++	-+	÷	+	++
\rightarrow	19	+	+-+-+	÷	<u> </u>	LL.		+	1	+	+-	: †•	-+-		+	÷	+	+			╞		+	÷-	-	-	+	╞		-+-	+		- <u> </u>		∔ -∔	╀	÷,	-+	+	t t		÷,	-	. •		+ - +	÷+	+	+ +	+	+	+	••-		,↑
	20	+	++-		• +	++	l i	+	+ +	+	+	++	+		+	+		+-	-	-			+	+	\vdash	+	+-	ļ		+	+	 	÷	+	łŧ	+	+		+	÷	+	+ +		+-+	-+-	•	·	-+	+ +	+	++	+	÷	1	t t
	21	4-	+	-+			┢┽	-+-	4-4		-+	÷	+-	+-+			+ +	-+	• •	+	+	$\left \right $	+	+			÷	÷	┞┼	-+-	+		÷	+	++	+	•	-	+-	+-+	•	÷	+	┽┥	-	- † ~-•	• •	+	+-+	+	+-+	-	÷+	-+	+-+
	22	-+-	÷-+	-	+ -	┼╌┽╌	+		+-	+		+ +	↓.		c	+	+ +	<u>,</u>	<u>+-</u> +	-+-	+-	H	+-	$+_I$		1	\pm	\mathbf{h}	+		'n	0	÷	÷	ŧŧ	-	+-+	÷	+	÷ -;	÷	·• ·1	+	+-1	- 1	÷	+ ·+·	t	+ †	ł	•	÷	1 · *	1	ŧŧ
	23	+	++-		• -•	÷÷	+-+		+-	-+		┝		+	2	'n	9	I e	÷	5,1	'la	5	5	0	e	TS	11	1	╉╌┤		11	C :	•	÷	łł	╈	÷	1	1	<u>+-</u> +	+	1 1	+	+ +			÷+	-+-	÷۲	+	+ +	. <u>.</u>	÷	. i	<u>+ '</u>
	24	+	++-	-	 i	+÷-	+	Ļ	÷	+		÷	-+-	┢			H		┝┥		t		+	+	+	Ŧ	10	-	h	+	10	÷ +	+	1	ł	-	ta i		t a	+++	a	+		In	PC	<u>.</u>	ŧ•-	-+	÷-+	+	++		++-	+	+-+
	25	+	++-	÷	÷+	++	ł	-+-	+	+	Ť	÷	-+-	+		<u> </u>	0	<u>i e</u>	† t	511	19	6	E		e	Ψ	N C	C	₽4		10	44	-		+ †		1	r +		+ 1	4		֠	ť	-+	+•	+ +	-+	+ †	+-	† •	-	+•	+	+ -+
	20	+	+.+	• +	<u></u> +	1 1	+	+	÷		÷	† •		╀	D.	- :		+	h		1	ł:†		te	+		n	6	IJ	0	rv	<u>+-</u> +.		0	þ		f	⊢	+ + h	þ	ir	10	p	ir!	+		• •	+-	+ !	+			<u>+</u> •	-	÷+
	21	-+-	$+\dot{+}$	++-	+	<u>∔</u> .	÷	+	+-	H	+	÷		+	+		11	1;	-11		10	⊬	14	2 >	+		4	F	ľ	-	, r	• •	<u>, 4</u>	+3	+-	+	+	-+	±#!	+	÷	÷	Ť,	+	÷.	:+	• • •	+	- a n n a	-+-	+	+	∔ +-	· · · • · · -	+ -+
\rightarrow	10 20	+	÷+-	++-	<u>+</u> •	• †	\uparrow	++	+	††	+			+	11	C P	÷+	7 0	+		ti	c	k	-	t	0	e	M	Þ	h	5 4	+++	7 6		N	elv	i	0	n	H	a r	d	H	n	ď	N	2	W	0	n	0	rd	e	r.	Ħ
	20	H	++	÷-+-	÷	֠	╈	÷	+	÷		÷,	+	+	Ч.		++	0,3	+		ť	٣		-	+	ř	1	1	Ŧ.	<u>.</u>			÷		ť,	Ť		<u> </u>	+			+	ا ۲			1	++	+	÷-•	1	Ť		† •		++
	30			.		<u>i </u>	<u> </u>							+	L			· ·	÷	1	<u> </u>	1 1			<u> </u>	1			<u>+</u>				_		÷																				

Job Specifications







All three types of records in the input file must be described. Because they are to be arranged in a certain sequence in the record-type group, they are described in the order they will be read. The first record in the group is the item master record (identified by M in position 1). The 01 in positions 15-16 indicates that this record is first. The record is required (position 18 is blank) and there must be only one per group (1 in position 17). It is assigned record identifying indicator 10. All fields found in the record are then described. Note that the record code appears as a field on the record layout form. However, on the Input Specifications form it is not described as a field but as the record identification code.

The second record in the group (02 in positions 15-16) is the issue record, identified by I in position 1. There may be several of these records per group (N in position 17). The record is optional (O in position 18). The record identifying indicator 20 is assigned. Fields in the record are then described. The third record in the group is described as were the previous two.

ITEMNO field in the item master record is assigned as the control field. When it changes, all transaction records for the item number have been processed.

RPG CALCULATION SPECIFICATIONS



	Unter	1 m	~		1				raante.		$\mathbf{F}(1 > 2)$	2 1 < 2 1 = 2	
		111	2		2					CIT	Lunk	up(Factor 21is	
		18 3	5 2	2	ž					ŏ.	🛱 Higi	Low Equal	
	3 4 5	11 7	8 9 10 11	42 1 14	15 11 17	18 19 20 21 22 23 24 25 26 27	28.29.30.01.52	33 34 35 36 37 38 39 40 41 42	43 44 45 45 47 48	49 50 51 52 5	3 54 5	5 56 57 58 50	60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
	0 1	c	20	1		ISSUES	ADD	TOTISU	TOTISU	50			FIND TOTAL SOLD
	0.2	c	39	\$		RECEET	ADD	TOTREC	TOTREC	50			FIND TOTAL RECD
	0 3	сZ	1			ONHAND	SUB	TOTISU	YEWONH	60			FIND NEWTOTAL
	0 4	c L	1			NEWOWH	ADD	TOTREC	NEWONH				OMH AMD
	0 5	c 🖌	1			ONORD	SUB	TOTREC	NEWONO	60			FIND NEW ONORDR
	0 6	c 🖌	1			TOTISU	SUB	TOTISU	TOTISU				CLEAR TOTISU
	0.7	c Z	1			TOTREC	SUB	TOTREC	TOTREC				CLEAR TOTREC
	0.8	c											
	0.9	с											
1	- T. 1			[¹¹]						E 8 J 4 1	1	1 1 1 1 1	

To update the quantity on hand and on order, total number issued (TOTISU) and total number received (TOTREC) for each item is needed. Quantity sold is found only in the issue record. Thus, the calculation to find TOTISU is done only when the issue record is read. Record identifying indicator 20 was assigned to the issue record. When 20 is on (an issue record has been read), we can calculate TOTISU. Thus, the operation (line 01) is conditioned by indicator 20. The operation to find TOTREC can be done only when a receipt record is read. The operation (line 02) is conditioned by 30, the record identifying indicator assigned to the receipt record.

Calculations to update quantity on hand and on order are total operations and can be done only after all transaction records for the item have been processed. They are conditioned by L1 which is set on when a new item number is read.

The operations on line 06 and 07 must be used to clear the total issues (TOTISU) and total receipts (TOTREC) fields after the quantity on hand and on order have been calculated.

IBM Tensor categorial Basines - Adactiones Composition

C

Program Stock Status

Programmer John Doe Date 1/10/--

Laborators

۸n» ا

IBM	i toterr	national Business Machines Corporation		RPG	OUTPUT	SPECIF	ICATIONS		GX21-9090 UM/050* Printed in U.S.A.
Program Program	S	tock Status Tohn Doe 🔤	0/	Keying C Instruction F	Graphic Key		Card Electro Number	Page 4 of Iden	gram 5 76 77 78 79 80 http://www.com/com/com/com/com/com/com/com/com/com/
Line	n Type	Filename	Skip Outpu	t Indicators	Field Name	End Pesstein In Context	Yus Yus No No No No No	Hances Nu Sign CR - X - int A J V 2 B K 7 3 C I Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z	Premove Plus Sign Date Fried Edit Zerio Suppores
3 4	0 9 For	$\frac{ }{ } \frac{ }{ } \frac{ }{ } \frac{ }{ } \frac{ }{ $	* 20 21 72 23 24 75 74 6 1 <i>P</i>	₹ 27 28 28 30 41	*AUTO	40 41 42 43 44	45 46 47 48 49 50 11 62 50 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Constant or Edit Word	20 21 22 23 24 / 55 66 67 68 69 70 21 72 23 74
03	0					59	STOCK S	TATUS REPO	אזיין איז איז איז איז איז איז איז איז איז איז
0 6 0 7 0 8 0 9 1 0	0 0 0 0		10			9 28 56 71	'I'TEM NO' DESCRIPT QUANTITY TRANSACT	ION' ION', QUANTIT	Y''
1 1 1 2 1 3 1 4	0 0 0	0K 0 1	0V			56 68	`ON_HAND `QUANT_ITY	, ON ORDE	R
1 5 1 6 1 7 1 8 1 9	0 0 0 0		200		ITEMWO DESC OWHAND1 OWORD1	8 34 43 56			
2 0 a 1 0 1	0	O1	3ø		<i>ISSUES</i> a	17 68	ISSUE	Page 95 of Program	75 76 77 78 79 80 ication 570KS7
0 2 0 3 0 4 0 5 0 6 0 7	0 0 0 0 0 0 0 0	7 72			RECEETQ NEWONH 1	19 68 43 45	` <i>RECEIPT'</i> ` **		
0 8 0 9 1 0	0 0 0					58	***/		
1 2	0	Heading lines are to	o print on ev	ery page	. They are co	nditione	ed by 1P and O	√ used in an OR re	lationship.
1 4 1 5 1 6 1 / 1 8	0 0 0 0	Not all detail lines is read. It is condit record. The second third detail line prin	are to print tioned by 10 d detail line nts when a r	for each), which i prints wh eceipt re	record. The f is the record i nen an issue re cord is read.	first shou dentifyir ecord is r It is con	uld be printed w ng indicator assi read, so it is con ditioned by 30.	when the item mast igned to the item r iditioned by 20. T	rer record
1 9 2 0		The printer spacing These words are no asterisks indicating	g chart show ot fields so t totals are al	s that the hey are e lso entere	e words ISSU intered as con ed as constant	E and R stants or is on the	ECEIPT are to p the appropriat total line.	print on the detail te detail lines. Not	lines.
7/ 1/ 0.	0	9 49 99 69 69 69 69 69 69 69 69 69 69 69 69	16 05 55 75 19 DG 60	80 / b 9t 5t bt	10 Z = 12 D2 6E 99 ZS 3€		06 Fri 82 12 92 52 52 1	5. 02.61 BL 21.91 51 91 51 21 11 4	

RESULTING INDICATORS

Sometimes your decision to do a certain operation is based on the result of a previous operation. Resulting indicators allow you to specify which operations you want done and the conditions under which the operations are to be done. Resulting indicators can be used to determine such things as:

- Whether a result is larger, smaller, or equal to a predetermined number.
- · Whether a certain result is plus, minus, or zero.

Program Cycle Operations

Figure 19 shows the operations in the program cycle associated with resulting indicators. Resulting indicators are set when the associated calculation operation is performed. This means that resulting indicators can be set either at detail or at total calculation time.

Resulting indicators are not set off automatically. They change their setting only at the time a calculation is performed or when they are set off intentionally. For example, if a resulting indicator is set on by a detail calculation, it retains this setting until the next time it is used as a resulting indicator.



Figure 19. Program Cycle Operations for Resulting Indicators

RPG II Specifications

The type of operation used to check the result field depends on the type of result being checked. If you want to determine whether the result field is larger, smaller, or equal to a certain number, you must use a compare (COMP) operation. If you want to determine if the result field is plus, minus, or zero, use an arithmetic operation (ADD, SUB, MULT, DIV). You can specify resulting indicators 01-99 on these specifications forms:



Resulting indicators are assigned at (1), then used to condition calculation operations at (2) and output operations at (3).

Resulting indicators and record identifying indicators are used in the same way to condition output and calculation specifications. You should not use the same entry as both a resulting indicator and a record identifying indicator.

Using the Compare Operation

In many jobs you need to know whether a field is greater than, smaller than, or equal to another field. RPG II language has an operation code, COMP, which allows you to compare fields. The compare operation requires entries in these columns on the Calculation Specifications form:



Factor 1 (either a field name or constant)

- Factor 2 (either a field name or constant)
- Resulting indicators

When compared, Factor 1 and Factor 2 can be in one of three relationships:

- Factor 1 can be greater than Factor 2.
- Factor 1 can be less than Factor 2.
- Factor 1 can be equal to Factor 2.

You indicate that a test should be made to check for one, two, or all three of these relationships by entering indicators in the appropriate columns:

	Resulting								
	Indicators					÷			
	Arithmetic								
820	the follows Za	•			~~				
	Compare				~~~		·	¥.,,,	a
귀하는	223<21	2							
8 E E E								1.00	14
8 8 6	100 CC				. A A .				
2 60 8		olas.	1.EL	83 64	65 95	17 B	e en 7	0 74 7	7 73 74
							TT	TT	TT
			ļ	<u>↓</u>	₩-		4	4.4	
	1){(2)}(3	Л				11		11	
			7	ht	17	11	+ +	\dagger	++
							11		
			Т			П	ТГ		
	+ + +	-	-	⊷		++	4-4-	-	-
									11
	1	111		t t	\mathbf{t}	tt	++	++	++
						11	1 1		

- A resulting indicator entered in positions 54-55 tells the system to determine if Factor 1 is greater than Factor 2.
- A resulting indicator entered in positions 56-57 tells the system to determine if Factor 1 is less than Factor 2.
- A resulting indicator entered in positions 58-59 tells the system to determine if Factor 1 is the same as Factor 2.

The test you specify is made each time the COMP operation is executed. However, the resulting indicator is set on only when the proper relationship exists. If you entered indicator 50 in positions 54-55 to test whether Factor 1 is greater than Factor 2, indicator 50 would be set on only when Factor 1 is greater than Factor 2. If the relationship does not exist, the indicator is set off. When testing for more than one condition, you can use the same or different indicators in these positions. If you intend to do different operations for each of the three conditions, enter a different resulting indicator to test for each condition on the Calculation Specifications form:




If you want to do the same operations when either one of two conditions exists (Factor 1 is greater than Factor 2; Factor 1 equals Factor 2), you could use the same indicator to test for both conditions on the Calculation Specifications form:

We explained previously that constants can be used in arithmetic calculation operations, but they must be numeric constants. In a COMP operation, however, constants can be either alphameric or numeric. Rules for using alphameric constants as Factor 1 or Factor 2 are a little different from those for using numeric constants:

Rules for Numeric Constants	Rules for Alphameric Constants
 A numeric constant can be any combination of digits 0-9. Decimal points and signs can also be included. 	 An alphameric constant can be any combination of characters (letters, digits, and special characters). Blanks are also valid.
 The maximum length of a numeric constant is 10 characters, including sign and decimal point. 	 The maximum length of an alpha- meric constant is 8 characters.
 Numeric constants must not be enclosed in apostrophes ('). 	 Alphameric constants must be enclosed in apostrophes (').

When you use the COMP operation code, remember to always compare two numeric fields or constants or two alphameric fields or constants. You cannot compare a numeric field or constant to an alphameric field or constant.

Using an Arithmetic Operation

You can test the results of an arithmetic operation (ADD, SUB, MULT, DIV) for plus, minus, or zero by entering resulting indicators in the appropriate positions on the Calculation Specifications form:



- A resulting indicator entered in positions 54-55 tells the system to determine if the result field is positive (plus).
- A resulting indicator entered in positions 56-57 tells the system to determine if the result field is negative (minus).
- A resulting indicator entered in positions 58-59 tells the system to determine if the result field is zero.

The tests you indicate are performed each time the operation is executed. However, the assigned indicator is set on only if the field satisfies the condition tested. If you entered indicator 99 in positions 54-55 to test the result field for plus, indicator 99 would be set on only if the result field were plus. If the condition is not met, the indicator is set off. Again, as with the COMP operation, you can test for one, two, or all three conditions at the same time. When testing for more than one condition, you can use the same or different indicators in these positions. If you intend to do different operations for each of the three conditions, enter a different resulting indicator to test for each condition:





If you want to do the same operations when the result field meets either one of two conditions (plus or zero, minus or zero), you could use the same indicator to test for both:

JOB 6 (STOKST): USING RESULTING INDICATORS TO TEST CONTENTS OF RESULT FIELDS

Job Definition

Print a stock status report similar to the one in Job 5. The only difference is the addition of maximum and minimum balances. Item master records usually include the maximum and minimum on-hand quantity for all items. These figures are kept so that checks can be made, whenever the inventory is updated, to determine if quantity on hand is within the limits set.

The first line for each item in the report shows standard descriptive data for the item: item number, item description, quantity on hand, quantity on order, maximum and minimum balances. Subsequent lines give the detail on current transactions involving the item. Quantities remaining on hand and on order are calculated for each item and printed after all transactions for the item are listed. Whenever shipments reduce stock on hand below the predetermined minimum balance or whenever receipts push the quantity on hand above the predetermined maximum, an exception condition is noted on the report.

Job Requirements

Input: An inventory file consisting of three different record types. Formats of the three record types are:

Name_ITEM MASTER RECORD

CODE=M	ITEM NUMBER	DESCRIPTION	UNIT	QUANTITY ON HAN D	QUANTITY ON ORDER	MAX BAL	MIN BAL		f 1 F .
F	2 3 4 5 6 7	8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29	30 31 32 33	4 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

Two decimal positions

Name ISSUE RECORD



1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60

Name RECEIPT RECORD

CODE= R	ltem Number	Quantity Received	
	2 3 4 5 6	7 8 9 10 11 12	13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60

The file is organized in ascending order by item number. For each item, one master record is required. Issue and receipt records are optional. When present, however, there may be any number of each. Records for each item are in this order:

- 1. Item master
- 2. Issue(s)
- 3. Receipt(s)

Processing:

- Find total number of each item sold. To do this, perform the calculation ISSUE + TOTAL ISSUE = TOTAL ISSUE for each issue record.
- Find total number of each item received. To do this, perform the calculation RECEIPT + TOTAL RECEIPT = TOTAL RECEIPT for each receipt record.
- When all transaction records for one item have been read, find new quantity on hand (ON HAND + TOTAL RECEIPT - TOTAL ISSUE = NEW ON HAND) and new quantity on order (ON ORDER - TOTAL RECEIPT = NEW ON ORDER).
- Compare the new quantity on hand to maximum and minimum balances to determine if an exception condition should be noted on the report.

Output: A printed stock status report:

		STOCK S	TATUS REPORT			
ITEM NO	DESCRIPTION .	QUANTITY ON HAND	QUANTITY ON ORDER	TRANSACTION QUANTITY	MIN. BAL.	MAX. BAL.
411116	B500 TWIN SOCKET BLUE 'ISSUE Receipt	458	500	50 500	800	1600
		908**	0**			
411122	B506 SOCKET ADAPT BRN ISSUE ISSUE ISSUE	325	0	20 38 10	300	900
		257**) * *		UNDÉR	
411173	C151C SIL SWITCH IVORY Receipt	50	150	150	100	200
		200**	000			
411254	A210 PULL CORD GOLD Issue Issue	62	75	16 30	60	165
		16**	75**		UNDER	

This printer spacing chart shows the format of the report:



RPG II Programming Language 107

Job Specifications



RPG CALCULATION SPECIFICATIONS

.

GX21 9093-2 UM/050* Printed in UIS A

IBM International Burn of Machines Cons	ากสาย	G CALCOLATION SPECIFICATIONS										
Program Stock Stð Programmer John Doe	tUS Keyin Date 1/10/ Instruc	g Graphic	Card Electro Number	1 2 75 76 77 78 79 80 Page Ø3 of Henchcation STOKST ST ST								
C Indicators	Factor 1	Operation Factor 2	Result Field	Resulting Indicators Arithmetic Pils Mirmag Compare Compare 1 > 21 - 2 Lookvulf Extorn 21is High Equal								
$ \begin{array}{c} $	JISSUES F RECEET F ONHAND S NEWONH F ONORD S NEWONH C NEWONH C TOTISU S TOTREC S	DD TOTISU DD TOTISU DD TOTREC UB TOTISU DD TOTREC UB TOTREC OMP MAX OMP MIN UB TOTISU UB TOTISU UB TOTISU	TOTISU TOTISU TOTREC Ø NEWONH Ø NEWONH NEWONO 6 Ø TOTISU TOTREC	9 9 9 9 9 9 9 9 9 1 5 1 1 1 1 1 1 1 1 1 1 1 1 1								
1 3 C 1 1 4 C 1 1 5 C 1 1 5 C 1 1 6 C 1 1 7 C 1 1 7 C 1 1 8 C 1 1 9 C 1 2 0 C 1 C 1 C 1 C 1 C 1 C 1 C 1 C 1	Calculations on I hand and quanti After new quant to MAX to see if in positions 54-5 is greater than Fa On line 07, NEW than the minimu	ines 01-05, 08, and ty on order. See Jol ity on hand (NEWC it exceeds the max 5 specifies a test to actor 2 (MAX). If N ONH is compared t un set. If it is, indic	09 are needed to updat o 5 for an explanation o DNH) has been calculate imum limits set (line Of determine whether Fac lEWONH is greater, ind to MIN to see if quantity cator 88 is set on.	te quantity on of these entries. rd, it is compared 3). Indicator 99 otor 1 (NEWONH) icator 99 is set on. y on hand is less								



FIELD INDICATORS

Field indicators, like resulting indicators, are used to test the contents of a field and to condition operations based on the results of the test.

Program Cycle Operations

Figure 20 shows the operations in the program cycle associated with field indicators. Note that input fields are tested and field indicators are set to reflect the result of the test at the time data, is moved into the processing area. Field indicators are not set off at the end of the program cycle. If a field indicator is set on when data is moved into the processing area in the first cycle, it is not reset until the appropriate field is moved into the processing area in a subsequent cycle.



Figure 20. Program Cycle Operations for Field Indicators

RPG II Specifications

Make these RPG II specifications when you use field indicators:





Field indicators are assigned at (1). They may be used to condition calculation operations at (1) and output operations at (3).

You can enter any one of the indicators 01-99 in positions 65-70 on the Input Specifications form to test an input field. You may assign indicators to test for three possible conditions:



- A field indicator assigned in positions 65-66 tells the system to determine if a numeric input field is positive (plus).
- A field indicator assigned in positions 67-68 tells the system to determine if a numeric input field is negative (minus).
- A field indicator assigned in positions 69-70 tells the system to determine if an alphameric input field is blank or a numeric field is zero.

JOB 7 (AGETB): USING FIELD INDICATORS TO TEST CONTENTS OF INPUT FIELDS

Job Definition

Create an aged-trial balance report that lists:

- Name and customer number of all charge customers who have payments due.
- Amount due.
- Overdue balances.

The customer master file, which contains records for all customers regardless of their balance, is used as the input file. The report is to show only those customers with payments due. Thus, information from customer records that contain a zero or credit balance should not be printed.

Job Requirements

Input: A customer master file consisting of one record type:



Processing: Check input balance due field for zero or credit balance (use field indicators).

Output: A printed aged-trial balance report:

r-

		AGEU TRIAL BALA	NCE				
CUSTOMER NUMBER	CUSTUMER NAME	LAST PAY DATE	CREDIT LIMIT	CURRENT CHARGES	00 30 DAYS	ERDUF ACCUU. 60 JAYS	214 2440 06
10867 16535 17849 18978	ALLEN & CO. ANDERSON AUTO SUPPLY ANDRENS AND SUNS INC	2/16/ 1/28/ 2/05/	15.000.00 2.500.00 750.00	6,919,77 1,665,49	375.58		10 2413
24743 25271	BERKLEY PAPER CU BERKLEY PAPER CU BEST DISTRIBUTION CU	12/27/ 2/21/ 10/06/	2+000+00 6+300+00 1+000+00	2,111.30, 1,185.50 3.25	611•54 2•652•45	145+54 312+13 1+400+05	90.4 51.0 767.1

This printer spacing chart shows how the report is formatted:

		4 4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5	2 3 9 3 3 6 8 6 6 6 5 6 7 8 9 0 1 2 3	6666667 4567890 / ALBAL			8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9) 1 1 1 1 1 1 1 1 1 1 9 0 0 0 0 0 0 0 0 0 0 0 9 0 1 2 3 4 5 6 7 6		1 1 1 1 2 2 2 2 4 3 6 7
CUSTOMER WYMBER XXXXX	eus toner NAME XXXXXXXXXXXXXXXXX	XXXXXXXX	4AST XX/:	PAY PAT	CREDI L/M/J XX,JXX.X	T. CURREN CHÂRGE X.X.,XXX.	7 5 30 XX X 2	0.v∈RDu 0.445 60. XX-XX-XX-	E ACCOUNTS ZAXS 90 DAX XXXXX XX,XXXJ	 s XX
\$.'nq	le spage	detail. line	s							-+++ ++++ ++++ ++++ ++++ ++++ ++++

Job Specifications





Conditioning Operations by More Than One Indicator

In this chapter, you have learned about many different kinds of conditioning indicators: control level, first page, overflow, last record, record identifying, resulting, and field indicators. In many jobs you will use two or more conditioning indicators. Indicators used together can be in either an OR or AND relationship.

You have already read about indicators in an OR relationship. You learned that if an operation can be done when either one of two conditions or both conditions exist, you can specify the conditioning indicators like this:





Conditioning indicators can be used in the OR relationship on both the Calculation Specifications form and the Output Specifications form.

If you specify two or more conditioning indicators on one line, they are in an AND relationship. The AND relationship means that all conditions must be satisfied before the operation will be performed:





If your calculation or output operation must be conditioned by more than three indicators, additional indicators can be specified on the next line if AND is entered in positions 14, 15, and 16 of the Output Specifications form or AN is entered in positions 7-8 of the Calculation Specifications form:



Five indicators used in an AND relationship condition this detail record. Three indicators are specified on one line; the remaining are specified on the following line with the word AND in positions 14-16. Indicators 10, 11, 12, 15, and 16 must all be on before the detail line will be printed.



This calculation operation is done only if indicators 25, 30, 31, and 90 are on and 91 is not on. Note that the operation is specified on the AN line.

Chapter 3. The Programmer's Job

Your responsibilities as a programmer include:

- · Determining the job requirements
- Determining what RPG II specifications and program cycle operations are needed for the job
- · Writing the specifications
- · Documenting the program
- · Preparing your source program for compilation
- · Compiling the source program
- · Testing the program

DETERMINING THE JOB REQUIREMENTS

The requirements for a job are generally described in terms of the input provided and the output required. The following paragraphs and illustrations describe the job requirements.

An invoice is to be prepared like that shown below:

	I	NVOICE		
ACCOUNT NUMBE	R 09621			
NAME	SMITH MANUFACTURING			
ADDRESS	L3620 9TH ST NE BERNALILLO NEW MEXICO 56120			
SHIPPING INST	RUCTIONS BY AIR DESCRIPTION	QUANTITY	UNIT PRICE	AMOUNT
439167	SHEARS	100	27.56	2,756.00
(]0(00	GASKET CORK	3000	1.15	3,450.00
029408				
102139	SPRIDGET WHITE	50	750.00	37,500.00
102139	SPRIDGET WHITE	50	750.00	37,500.00

The input file contains two types of records: name/address records for all customers who made purchases on credit during the month and transaction records for each item purchased by the customers during the month. The name/address and transaction records look like this:

Name NAME/ADDRESS RECORD

ACCOUNT NAME ADDRESS LINE 1 ADDRES	2		I
	NAME WUMBER	ADDRESS LINE 1	A DDRESS
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 44 35 36 37 38 39 40 41 40 42 44 45 46 45 45 45 46 45 46 45 45 45 46 45 45 45 45 45 46 45 45 45 45 45 45 45 45 45 45 45 45 45		6 27 28 29 30 31 22 33 34 35 35 37 38 30 40 40 40 40 40 40 40	

LINE 2 ADDRESS LINE 3 61 62 63 64 65 66 67 68 69 70 77 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96

Shipping code (either 1, 2, or 3)

Name TRANSACTION RECORD

= 7 =	ACCOUNT		ITEM	2		UNIT	
CODE	NUMBER		NUMBER	DESCRIPTION	QUANTITY	PRICE	
	2 3 4 5 6	78	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

The input file is organized so that all transaction records for a customer follow the customer's name/address record. Each customer has one name/address record, but might have one or more transaction records.



Standard computer paper is to be used for printing the invoices. Each invoice should be formatted like this:

		122222222223333333333		5 5 5 5 5 5 5 5 5 5	566666666	
1			1234567890	123456789	012345678	901234567890123456789012
2	╋ ╡╿╡┊╿┥╷┥┫╺┊╓╡┊┥	╸ ┿╋┾╄┿┵┽┿┿┽╋╊┿┿┽┿┿┿	╉┯╍┯┿┵┵╆┿┿	╉┽┽┵┾┿┽╺┽┽	┟┟┽┥┽┿┿┥	QU <u>U VLOQUAN</u> O <u>D</u> ODO <u>DDON</u> O A S
3		╸┥╊┥╷╷┊╷╷┊╷┊┥╡┨╎╎╎╷┥	┫╼┿╾┲╼╪╺╧┲╧	╉ ┊╸╸┊╸┧╷┊ ┊	╉┿┽┽╸╾╸┽╴	┥┼╊┽╤┽┼╺┽┽┼┽╉ ╦╌┥┿┿┥┫╶╝╸
H_ •			VOICE	╊┾┼┯╸┽┽┊┼┷	╉┾┼┼┾╍┯┽╴	┥┾┲┝╍┿┼╍┿┽┾┼┲┼┲┶┶┶╍┝┵╾┏┵╴
				┨╀┼┿╿┿┩┊┤╼	<u>╋</u> ╪╪╪╺╶┥┿╄┅	╞╌┣╄╹┼┥┥┝┊┥┽┫┊┝┥╺┝┥┥┥
			₱·+ + ►·+·+·+·+-+-+-	╅┿┯┿┿┿┿╺╺	╋┿┿┿╼┿╺┝	[╄] ╾╂┽ ┊╕╪╸┊┊┊╸┫┥┥┝┿╻┥┥┥ ╉
- 11 7	ACCOUNT NUMBER	XXXXX (ACCA/D)		1 *********	┫╷╽╽┑╻╷ ╽╴	<mark>╴╸╊╶┨╺╘╺┥╸┥╺┥╸┨╶┥╺┥╺╶┥┥┥┥</mark>
	┡╀┥┿┼┍┾┽╡╂╶╷┥┽┊ ╺╶	 ╊┼┼ ╌ ┼┼┼┼┼╎╷╷╷╷╷		_ • • • • • • • • • • •		┊╴┫╉╸┽┼╾┽┼┼┼┫┼┼┼┼┼┼┼┥┨╋┿┥
H 10		- With the state of the state o	· + 1 X 12 + 4 -			
11			(NAME).	╃┍╾ ┍┽╺┍┍┍	• + + + +	
H 12	ADDRESS	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	VY 71550	FECTIME		┝╼╂╀╍┾╪╍┷╧┾┅╂┽┾┲┾┥┥┥┥┛
_H 13	<u> 2282302222 22907977</u>		XX ZADDA	ASSI INF	⊭ _{+	┝╪╋┠┼╧┾╾╞╼┟╼╋┽╪┿┽┾┿┽╻┊┫╓┊
_ <u></u> H]14			XX (ADDR	ESSLINE	13 5 ⁺ · · · · −	┍┽╉╄ ┥ ┿┽╾╸╸┽ ╺╏╷┥╺╺┥┥┥ ┨┽┽
15	╺┾╾┥┈┵┽┼╍┼╺╴┠╼╶╸┼┥╺		• ₩ 4:	1777+771 7 -		╵┽┫┟┑╾┧╺┽┰╸╺┫╌╾┾╴┝┿╁┲┽┨┽╀
	╘╍┼┽┽┽┽┽┿╋┿╋┿┝┿┿	• + • • • • • • • • • • • • • • • • • •			*****	╘┿╉┦╍┥┯╍╸┶┲╕╉╍╍┊╍┽╸┼╞┊┨┽┼
	╘╪┼╄┿┥┶┿┥┾╋╸ ╁ ┾ ╪┽ _┥	┍╪╋╪╪┿╍╸╍╾┽┵╍╉╫┟┟╻┎╻╻╽╎				
H 19	SHIPPING INSTON	CTUDE WWWAAAA	1.0.0		L	
20			AIR, BY	TRUCK, 0	R.BY R	A/ 4_
H 21	ITEM MUMBER	DESCRIPTION	QUANTITY	· · · · · · · · · · · · · · · · · · ·		╶╸╂┼╸╷╸╸╻╸╸╕┨╝╸┼╼┍╸╸┥┼┠┥┿
22				-+ + +-+ + > / / / + / +-	FAILS .	AMO'UMT.
D 23	XXXXXXX	XXXXXXXXXXXXXXXX	XXXXX	······ XXX	. W	
- 24						
25	-++ \$.4 <i>1.</i> 4 . M <i>1.</i> Y2 <i>1</i>	\square \square \square \square \square \square \square \square \square \square	(QTY)	[UPA	PIGE	(AMOUNT)
27	╶╞┊<u>╞</u>╺╞┥┥┥┥┥	╶╾┫╷┽╾┍╼╍╼╍┊┟┽╾┽╺╌╸╸╸┊┊	+++++++++++++++++++++++++++++++++++++++			
T 28	╺╉┊┦┤┨┦╼┼┽╉╾┼┽┥┾╺╼╸	╶╷╊┋┇╸╸╸╸╸╸╋┥┥┥╸╸╸ ┽┥				
29		╶┼╋┼┼╍┼╍╍╍╼╋╆┶┶╍╼┶┊┫		INVOICE	IOTAL	XX 2 XX X 9 XX X • XX
30	. 222200000 00200000			· • • • • • • • • • •	┝╪╪╪╪╤╧╉╌╪	+++++++++++++++++++++++++++++++++++++++
31				· • • • • • • • • • • •		┿ <u>┧╋╱╱╃</u> ╨╸ <u>╚</u> ╞┥ <mark>┟</mark> ┥╴╺╺╸┥╸╸╸┨╶╴╴
32				···· + · · · · · · · · · · · · · · · ·	┝┿┿┯┽┿╺┥┥┝	╺╊┼╸╾┥╸╸┽╿╺┨┶┽┽┽┿┿┾┾┾╊┾┿
	+++++++++++++++++++++++++++++++++++++++	pre space de	Tall Th	nesj		╶ ┲┲ ╹╹╸╹╹╹╹╹╹╹╹╹
35	╹╋╪╋╋╞╪╪╪╪┍╞┨┍┥╡╞┊┥╸╸	┽╂┼┼╾╾╾╾╅┼╼┽╼╼┽╉				
++	╹╏╎╹╎╿╎╵╹╎┨╹┥┥┥┥	╶╪╋┤╾╾╾╸╸┥┥ ┨┊╸ _{┥╼╼╼} ╺╺		· · · · · · · · · · · · · · · · · · ·	+++++++++++++++++++++++++++++++++++++++	
			,	•		The second second second second second second second second second second second second second second second s

Your first step is to analyze the problem and decide what processing must be done to get the desired results. Always keep in mind how things are done using RPG II. In your analysis of the job, you would probably think of these points:

- Information for the first part of the invoice is taken from the name/address record. Information for the second part (list of transactions) is taken from transaction records.
- In order to print shipping instructions, the shipping code recorded in the record must be determined:

1 = By truck

2 = By rail

3 = By air

 AMOUNT and INVTOT (invoice total) must be calculated because this information is not in the input records. These calculations must be done for all transaction records:

QTY x PRICE = AMOUNT

AMOUNT + INVTOT = INVTOT

- INVTOT should be printed only after all transaction records for one account have been processed.
- The invoice for each customer must be on a separate page. This means that forms must advance each time a new customer name/address record is found. It is possible that one customer has purchased so many items that they cannot be listed on one page. In this case, forms should advance when the end of a page is reached. When an invoice includes more than one page, headings should be printed on all pages.

DETERMINE RPG II SPECIFICATIONS NEEDED FOR THE JOB

After you have carefully analyzed the job, determine what RPG II specifications and program cycle operations you need. For example, consider the following:

- Different record types are used. This means that record identifying indicators must be specified to tell what to do for each record.
- The shipping code must be determined. One way to do this is to compare the shipping code to 2. Through the use of resulting indicators, you can determine if the code is less than 2 (1), greater than 2 (3), or equal to 2.
- INVAMT is printed only after all transaction records for one account have been processed. This is a total operation, done only after a group of records has been processed. Therefore, control fields and control level indicators must be used to do a total operation. The account number field can be used as the control field.
- Forms should advance each time a different name/address record is encountered or whenever overflow occurs. Thus heading lines must be conditioned by a record identifying indicator and the OV indicator.

If the indicators and steps just listed are used, the RPG II program cycle would include the steps shown in Figure 21.



Figure 21. Program Cycle Operations for Sample Job

WRITE THE SPECIFICATIONS

After you have analyzed the problem and determined how to solve it using RPG II, you can write the specifications. Figure 22 shows the specifications for the job.



Figure 22 (Part 1 of 4). Specifications for Invoice Preparation Program







IBM	inte	mational Business Machin	us Corpo	oration			RPC	3	OUTPUT		SPEC	FI	CATIONS			GX21.96 Printed i	90 UM/ n.U.S.A.	050
Program	Ī	nvoice	Pro	epar	atio	n			Graphic T		· · · · · · · · · · · · · · · · · · ·		Card Electro Number	1 2		75 76	77 78 79	80
Program	mer	John Do	e	Date /	1101-	-	Neying Instructio		<ev< td=""><td>++</td><td></td><td>-+-</td><td>-</td><td>Page ϕ4 of</td><td>Program Identificatio</td><td>IW</td><td>VOC</td><td>E</td></ev<>	++		-+-	-	Page ϕ 4 of	Program Identificatio	IW	VOC	E
				1-1		·····	· · · · · · · · · · · · · · · · · · ·	l		. I I.	. I					··· E		
0				49 Space	Skip	Out	put Indica	tors						1. 1	r	····	1.25	
	ן ר		(T/E)				T		Field Name	11			Commas Commas to Prin	nces No Sign CR	L X · Remov Plus Si	e In		
Line		Filename	(H/D	Stkr.					There have	1		Π	Yes Yes Yes No	1 A 2 B	J Y Date K Field B	dit		
	8	i neaturite	Å,	Rate		A	nd And			t at	End Position		No Yes No No	3 C	L Z - Zero M Suppre	55		
	1 41			ADD	fore fter	8	8 8			odes	in Output	/8/			· ·	<u> </u>		
	Forr		OR		a A	ž	žž		*AUTO	Edit (Record	P/8/1	U.C.	onstant or Edit Word	1			4
3 4 1	5 6	7 8 9 10 11 12 13	14 15	16 12 18	19 20 21	22 23 24 25	26 27 28 29	30 31	32 33 34 36 36 37	38 39 4	10 41 42 43	44	1 2 3 4 5 6 7 8 9 15 46 47 48 49 50 51 52 53 54	10 11 12 13 14 15 16	17 18 19 20 21 2 67 63 64 65 66 6	2 23 24 /	78 22 85	
0 1	0	IMVOICE	h	3	Ø4	1Ø						Π						Π,
0 2	0		OR			jov	W1Ø					П						H
0 3	0	-+									45	Π	`INVOICE'					
0 4	0		H	3		10						Π			11.11			
0 5	0		OR			OV	W1Ø			П								Ħ
0 6	0										17		ACCOUNT N	UMBER		111		
0 7	0								ACCWO		23	Π						
8 0	0		H	a		10		1			110						-+-+-	
0 9	0		OR		ľ.	OV	NIO	1										
1 C	0						f f				7	11	WAME!					-
1 1	0								NAME		39			! ! ! !		+-+-+		
1 2	0		H	1		10		1								+ + + +	-	
1 3	0		OR			lov	NID	1			: 11							
14	0							1			10		ADDRESSI		+++++	+ + + +		-
1 5	0							1	ADDR1		42			╡┝┈┾┼┼┼┥╽╷				
16	0		H	1		10							<u>┤</u>	┼┼┼┼┼┼		┼┼┽╏		
1 7	0		OR			0V	NIØ							╄╌┫┥╏┥┥┥┥┥┥	╡┼┼┼┼	╡┤┼╞		
18	0								ADDRA		42		<u>╋</u> ╋╖╫┈ <u>┡╍╋</u> ┥┽╋┥╋╶╋╴	<u> </u>		┼┼┼┟		
19	0		H	2		10		11				i i i i i i i i i i i i i i i i i i i	╪╍╡┊╗╞╕╪╌╡┊╏╽╶┨╍	┼ ┥ ┈┼╶╀╌╃╶┥╺┤╺┤	+++++	┼┼┼╊		-
2 0	0		OR			OV	NIO						╪╌┥╀╴┥┍╋╌╸	╄ ╶╿╺╿╺╿╺╿╺╿╺╿	┽┽┼┽┼	┼┼┼╊		
	o											T		<u>┝╶╊╌┡╼╞╼</u> ┠╌┼╌╋	┿╫┼┿┿	┿┼┿╋		-
	o		Π										┊┥╪╌┿╌┥┝┈╉╾┾╼┾	┼ ┾ ┼┼┼┼ ┼┼┼	═┥╌┟╶┼╌┿╌┼╴	┼┼┼┣		
	0						111				+-+-+		╈╹╞╞╼╸╞╏┊	┼╌┾╌╎╴ ╏╶╎╴╞╸ ╈╴┿	╶╆╶┝╌╋╌┿╴	┼┼┤┠		-
	0										+-+-+	1	╪╾┥╴╄╶┾╌┿╌┽┈┽┄┟╼┼╼	<u> </u>	╶┼┼┾┽┼	┼┼┼╊		
	0							† † ·					+-+++++++++++++++++++++++++++++++++++++	╄╺ ┠╶┠╶╞╶┨ ┉ <u>╋╶</u> ╋╶┥	╶┾┽┤┼┼	┼┼┼╊		-
22 17 0	69	89 49 99 99 99 69 69 69	19 09	65 85 15	99 99 99 99 8	5 ZS 15 UH	61 81 /1 90	50 00	58 38 40 41 45 43	28 98 5	35.33.34.3	18 0	51 55 33 54 56 56 51 58 59 3	02 61 81 21 91 91 91 9		1, 9, 9		لې.

Figure 22 (Part 3 of 4). Specifications for Invoice Preparation Program



Figure 22 (Part 4 of 4). Specifications for Invoice Preparation Program

DOCUMENT THE PROGRAM

An important part of every programmer's job is to explain his program. This documentation provides information for people who will run the program and for programmers who may later need to alter or update it. Documentation is also useful to you. It is not always easy to remember what every program you wrote does. Reading documentation is a much easier way to recall the program than figuring out each instruction.

Documentation consists of:

- 1. Telling generally what the program does.
- 2. Describing input and output. (Record layout forms and printer spacing charts are an excellent means of describing input and output.) File names and field names should be meaningful.
- 3. Explaining the coding.
- 4. Telling the operator how to run the program, what to do if the system stops because of an error, and what to do when the job is completed.

All documentation cannot be done at the time you write specifications. However, when writing your specifications, you can also write an explanation for a line or lines of coding on the specifications forms. You have probably noticed positions labeled Comments on the specifications sheets. Here is where you write an explanation for your coding.

In addition to using the comment positions on the coding forms, you can use comment lines. A comment line is indicated by an * (asterisk) in position 7 of the coding form:

RPG INPUT SPECIFICATIONS GX21 9094 UM/05 IBM Unternational Business Machines Conjunction Printed in U.S.A. Printed in U.S.A.		
Program	Keying Graphic Card Electro Number	1 2 75 76 77 78 79 80
Programmer Date	Instruction Key	Page of Identification
	Record Identification Codes	Field Indicators
Line Filename E E E E E E E E E E E E E E E E E E E		reld Name
1 I #A COMMENT LINE IS	IMDICATED BY AN # IM COLUMN	A COMMENT LINE CAN
O 3 I #IWCLUDE ANY IWFOR	MATION YOU WANT.	
0 5 I K K K K K K K K K K K K K K K K K K	* * * * * * * * * * * * * *	* * * * * * * *

PG INPUT SPECIFICATIONS

Comment lines can be used anywhere on any specifications form. There is no limit to the number you can use. The RPG II compiler does not regard comments and comment lines as part of the program. Therefore, the compiler does not translate the comments into instructions; however, the lines are printed as part of the source program listing.

PREPARE FOR COMPILATION

After completing your source program, you must prepare it for compilation.

Specifications Form Order

Your specifications forms must be in this order:

- 1. Control and File Description Specifications form
- 2. Input Specifications form
- 3. Calculation Specifications form
- 4. Output Specifications form

Number the forms in positions 1 and 2. At this time, you might also check to see that the top part of each form is completely filled in.

If you are planning to give these specifications to someone to key, it is a good idea to fill in the box labeled Keying Instructions:

RPG	CONTROL	AND FILE DESCRIPTION SPECIFICATIONS
Program Programmer Done	Keying Instruction	Graphic Card Electro Number Punch

You indicate in this box the graphic symbols you are using and their meaning. Some printed letters and numbers are easily confused. For example, it is sometimes difficult to differentiate between the number 0 and the letter 0 and the number 2 and the letter Z. You may, therefore, devise a graphic symbol that you use for certain letters. Some people use \emptyset for zero, Z for the letter Z. Explain your symbols so that the operator will know what to key when the symbol appears on the coding forms.

Control Specifications Preparation

Some systems require control specifications. If yours does, you will have to fill out the control specifications at the top of the Control and File Description Specifications form.

Control specifications give the compiler information about the system and tell whether any special RPG II functions are used in the program. The entries are such things as:



Checking the Specifications

Desk checking is a good way to reduce the number of potential program errors. Desk checking means carefully checking through your specifications to see whether you have:

- Placed entries in appropriate positions
- Used correct entries in positions
- Spelled the same field and file names identically throughout your program
- Used your indicators correctly

If you should find that you have omitted a specification (did not name an input field or an output field or did not enter a calculation), you can enter it on a line following line 20 (line 10 on the Control and File Description Specifications form under File Description).

Notice that no line numbers have been entered in positions 3-5 of the specification lines located below line 20. You can place numbers in these positions to tell where the missing specification belongs:



If your specifications are being keypunched, the out-of-order cards must be inserted in the appropriate place. If the source program is being entered directly into the system through a keyboard, the missing specifications will have to be inserted in the appropriate place when the specifications are keyed.
COMPILE THE SOURCE PROGRAM

When you think your source program is free of errors, it can be keyed according to the source entry method for your system. You can then compile your source program. The important part of compilation is, of course, translating the source program into machine language. But in addition to this, the compiler also produces the following program listing:

		0001	н о	14							INVOCE
(0001	0003									
	RG 025	0002	FNAMAUU	TP	12	8 1 2 8			DSIK		I NV OC F
	0002	0003	FINVOICE	εJ	12	0 120		ΟV	PRINTÉR		I NV OC E
	0003	0004	INAMADD	011	10						
	0004	0005	I						2	60ACCNÚ L1	INVOCE
	0005	0006	I						7	26 NAME	INVOCE
	0007	0007	I I						27	49 ADDR1	I NV OC E
	0008	0009	Ī						73	72 AUUR2 95 ADDR3	
	0009	0010	I						96	960SHPCD	INVOCE
	0010	0011	I	021	20	1 CT					INVOCE
	0012	0012	i T						2	60ACEND L1	INVOCE
	0013	0014	I						15	29 DESCRP	
	0014	0015	I						30	340 JTY	INVOCE
	0015	0016	I						35	392UPRICE	INVOCE
	0016	0017	C 10		SHPCD	c	OMP	2		979899IS CODE 1.2.3	3•?INVOCE
	0017	0018	C 20		QTY	M	ULT	UPRICE	AMOUNT	52 FIND ITEM TOT	TALINVOCE
	0018	0019	ι 20		AMUUN	i a	00	INVIOT	INVTOT	102 FIND INV TOTA	AL INVOCE
	0019	0020	OINVOICE	н	304	10					I NV OC F
A I	0020	0021	0	OR		01010					INVOCE
\sim 1	0022	0022	0	н	r	10			45 'INVC	DICE.	I NV OC E
	0023	0024	Ō	OR	2	DVN10					
	0024	0025	0						17 • ACCU	JUNT NUMBER.	INVOCE
	0025	0026	0		-	•	A	CCNO	23		I NV OC F
	0020	0027	0	08	2						I NV OC F
	0028	0029	0	U.		00010			7 • NAME		
	0029	0030	0				Ν	IAME	39	-	INVOCE
	0030	0031	0	H	1	10					I NV OC F
	0032	0033	0	UN		OVINIU				FSSI	
	0033	0034	0				A	DDR1	42		INVOCE
	0034	0035	0	H	1	10					INVOCE
	0036	0037	0	UK		UNIO	۸	0082	4.2		INVOCE
	0037	0038	0	н	2	10	~	DONE	42		INVOCE
	0038	0039	n	<u>ne</u>							
	0039	0040	0	51		UN NLU	۸	DDR 3	42		
	0040	0041	0	Н 3	2	10					INVOLE
	0041	0042	0	OR		OANTO			a		INVOCE
	0043	0044	0				97		24 'SHIP	PING INSTRUCTIONS	INVOCE
	0044	0045	0				98		33 BY T		INVULE
	0045	0046	0		-		99		33 BY R	AIL.	INVOCE
``	0046	0047	U	н	2	10					INVOCE
-											

The Programmer's Job 137



The most important parts of the program listing¹ are:

A printout of source specifications including comment lines. Notice the number at the left of each line. This is the sequence number the compiler assigns to the specification.

B) A list of all indicators used in the program.

A list of all fields used in the program. Included in the list is the storage location assigned to each field and a description of each field as indicated in your specifications. Some compilers also provide this information for constants used in the program.

D Diagnostic messages indicating the types of errors made and the statement in which they occur.

¹Note that the sample listing shown is the program listing for the invoice job (INVOCE).

If the compiler finds any errors in your source specifications, it will print diagnostic messages telling you what errors were made. You will find that different types of messages are printed: warning, terminal, or informative. A warning message is an indication that something may be wrong. If you check the questioned specification and find that it is all right for your program, you need not make changes. If you get a terminal message, however, something is wrong with your coding. You must fix the specification and recompile the program before the compiler will actually translate your specifications.

The diagnostic message section of the program listing contains two basic parts: a list of messages (X) and an explanation of each message (Y).



Each error message in the list is identified by a 3-digit number **1**. Next to the message number is either a statement number identifying the specification in which the error appears or a field name or constant associated with the error **2**. Following the list of messages is an explanation of each error **3** and an indication of the severity of each error (W = warning; T = terminal) **4**.

The sample shown above shows diagnostic messages printed for the invoice job. Note that message 221 is a warning. A warning is an indication that something may be wrong. If you check the specification noted and find that it will work for the job, you need not change it.

Checking the message in the listing, you would find that the warning points to the AMOUNT field in statement 0017:

0017	0018 C	20	QTY	MULT UPRICE	AMOUNT	52	FIND ITEM TOTALINVOCE

The AMOUNT field is specified as five characters with two decimal positions. In checking your specifications, you will find that the amount should have been 7 digits with 2 decimals, and it was incorrectly keyed in the source data.

Message 0025 is a terminal error. This error number appears below the statement at which the error occurred. Checking the listing where it appears at statement 1, the device entry DISK, was miskeyed as DSIK. You must correct this error. (Check the RPG II reference manual for your system for the correct entries for the Device positions.)

TEST THE PROGRAM

It is good practice to test your program before using it for an actual job. To do this, make up test data representing all possible situations that could arise during an actual job. Run your program using that data to see if your program will really handle the situations your think it does. If you get the wrong results when testing, you know your program is not doing what you thought it would. You can usually find your errors by using actual input data and doing the operations specified yourself, step-by-step, in the order the system would do them. When doing this, you have to follow closely your specifications and the program cycle operations taken by your program. After you test your program and the results show it can handle all situations, your job is complete. address: A number identifying a location in storage.

alphabetic: In general usage, any of the letters A-Z. In RPG II programming, any of the letters A-Z and special characters @, #, and \$.

alphameric: Any alphabetic or numeric characters.

alphameric constants: Any of the characters in the data character set.

alphameric field: A field that contains any of the characters in the data character set.

AND relationship: The specifying of conditioning indicators such that the operation conditioned will be performed only when all conditions are met.

arithmetic/logic unit: An area inside the processing unit where calculations are performed.

arithmetic operation: An operation such as addition, subtraction, multiplication, and division performed in the processing unit.

ascending order: The arrangement of records in a file from low to high, based on the contents of a specified field in each record.

blank after: Changing the contents of a field so that it contains only zeros or blanks after that field has been written to the output record.

Calculation Specifications form: An RPG II coding form which specifies the type and order of calculations to be performed on the input data.

card: In data processing, a card containing combinations of holes representing data to a system.

card file: A group of related punched-card records.

card layout form: A chart for planning the design and format of cards.

card punch: A device that records information on a card in the form of combinations of holes representing characters.

card reader: A device that electronically senses information on punched cards and transfers that information to the processing unit.

character: Any individual data item that can be represented in printed form; that is, a letter, a digit, or a special character.

coding: Making entries on RPG II specification forms.

comments: Words or statements in a program that serve as documentation rather than as instructions to the compiler.

compile: Translate a source program (such as RPG II specifications) into an object program (machine language program) using the compiler.

compiler: A program that translates a source program into a machine language program.

conditioning: Using indicators to control when calculations or output operations are done.

constant: A data item that does not change during execution of a program. This item represents itself and is actually used in processing rather than being a field name representing the data. For example, COST is a name representing a field containing data that changes, whereas the constant 100 is actual data used that does not change.

control break: A change in the contents of a control field.

Control and File Description Specifications form: An RPG II coding form that gives, for a particular job, information needed for control of the system and a description of the files used.

control field: One or more fields that are compared from record to record to determine when certain operations should be performed.

control group: A set of records all having the same control field information.

control level indicator: An indicator used to specify certain fields as control fields and to indicate which operations to perform at total time.

control unit: An area inside the processing unit that determines from instructions what has to be done. It directs other units or devices to perform the required functions.

data: A collection of facts, numbers, letters, and symbols that can be processed or produced by a system.

data character set: All of the 256 EBCDIC characters.

descending order: The arrangement of records in a file from high to low, based on the contents of a specific field in each record.

detail record: An output record produced during the detail output operation of the RPG II program cycle.

detail time: An operation in the RPG II program cycle in which calculation and output operations are performed for each record read.

diagnostic message: An output message that identifies RPG II specification errors and their severity.

digit: One of the characters 0-9.

disk: A flat, circular plate with a magnetic surface on which data can be stored.

disk drive: A device that reads data from or writes data on a disk.

Disk file: A group of related records stored on disk.

diskette: A thin, flexible magnetic disk permanently enclosed in a semirigid protective jacket.

documentation: A written explanation of a program, its use, function, and operations.

edit: To punctuate a numeric field by suppressing zeros and inserting commas, decimal points, dollar signs, or other constant information.

edit code: A number or letter indicating that editing of a numeric field should be done according to a predefined pattern. This includes zero suppression and punctuation. eighty-column card: A punch card with 80 vertical columns representing 80 characters.

end of file: The end of records in a file.

error message: See diagnostic message.

execute: To process input data files according to machine language instructions to produce the desired output.

factor: In RPG II programming, a field name or constant used in a calculation operation.

field: One or more adjacent record positions which contain related information.

field indicator: An indicator used to show whether a given field in an input record is plus, minus, zero, or blank.

field length: The number of positions allowed for a given field, determined by the maximum length of information that will be entered in the field.

field name: In RPG II programming, a combination of no more than six alphabetic or numeric characters (the first of which must be alphabetic) that identifies a field.

file: An organized collection of related records.

file name: In RPG II programming, a combination of no more than eight alphabetic or numeric characters (the first of which must be alphabetic) that identifies a file.

first page indicator: An indicator used to specify which lines (such as headings) should be printed on the first page only.

half adjust: A method of rounding off a number by adjusting the last digit to be kept. When the number to the right of the last digit to be retained is 5 or greater, 1 is added to the last retained digit. For example, 2.475 half adjusted to two decimal places becomes 2.48, but 2.474 becomes 2.47.

heading: A constant, usually printed at the top of a page, identifying the information or report on that page.

indicator: (1) A 2-digit or 2-character entry on the RPG II specification forms used to tell when certain calculation or output operations are to be performed. (2) An internal switch used by the object program to remember when a certain event occurs.

input: Data that is to be operated on (processed) by the system.

input file: A set of records a program uses as a source of data.

input specifications form: A coding form used to identify the different types of records in each input file and to describe the fields in each record.

instruction: A statement that specifies an operation to be performed by the system and the locations in storage of all data involved in that operation.

Keyboard: A device used to enter data into a system.

Keypunch: A device, similar to a typewriter, used for punching information into cards.

last record indicator: An indicator that signifies when the last data record has been processed.

machine language: A language that can be interpreted and used by a system.

ninety-six column card: A punch card with 96 vertical columns representing 96 characters. The columns are divided horizontally into thirds, such that the columns in the upper third are numbered 1-32, in the middle third, 33-64, and in the lower third, 65-96.

numeric: Any combination of the digits 0-9.

numeric constant: A constant used to represent a number than can consist of a decimal point, a sign, and the digits 0-9.

object program: A set of instructions in machine language. The object program is produced by the compiler from the source program.

operation: A defined action performed on one or more data items, such as adding, multiplying or comparing.

operation code: A word or abbreviation specified on the Calculation Specifications form to identify an operation such as, SUB for subtract or ADD for addition.

OR relationship: Specifying conditioning indicators so that the operation conditioned is performed when either one or both of the conditions are met.

output: Data transferred from storage to an external medium such as printed form, punched cards, or disk.

output file: A set of records that is written, punched, or printed by the system to an external medium.

Output Specifications form: An RPG II coding form used to specify the records to be written in each output file and the format of the records.

overflow: The condition that occurs when the last line to be printed on the page has been passed.

overflow indicator: An indicator that signifies when the last line on a page has been printed or passed. It can be used to specify which lines are to be printed on the next page.

overflow line: The line specified as the last line printed on a page.

overflow page: The new page after an overflow has occured.

primary file: The main file from which a program first reads records.

printer: An output device that records information on paper in the form of printed characters.

printer spacing chart: A form used to plan the location of data in the printer output file.

processing: To perform operations on data from an input record.

processing unit: The part of a system that controls the system and its attached devices, provides storage area for the programs and data, and performs the operations specified in the program.

Program: A set of instructions that (when stored) tells the system which operations are to be done and how to do them.

program cycle: A series of operations performed by the system for each record read.

program listing: A printout which gives information about the source program, such as source statements, diagnostic messages, indicators used, storage addresses of fields, and constants used.

punched card: See card.

record: A group of related fields or data items treated as a unit.

record identification code: A code placed in a record when it is created to identify that record type.

record identifying indicator: An indicator that identifies the type of record being processed during the current program cycle.

record length: The total number of positions in a record.

record types: The classification of records in a file. Records are classified according to a specific field or fields within each record. Records of the same type have the same fields in the same order and identical record identification codes.

result field: The name of a field where the outcome of arithmetic calculations is kept.

resulting indicators: An indicator that can signify (1) whether the result of a calculation is plus, minus, zero, or blank, or (2) whether a field is greater than, less than, or equal to another field.

secondary file: Any file other than the primary file used in multifile processing.

Source program: A set of instructions representing a particular job as defined by the programmer. These instructions are written in a programming language, such as RPG II.

Special character: A character other than a digit, letter, or @, #, and \$. For example, *, +, and % are special characters.

Specification forms: Forms on which an RPG II program is coded and described. The four specification forms described in this manual are the Control and File Description Specifications form, the Input Specifications form, the Calculation Specifications form, and the Output Specifications form.

Storage unit: An area inside the processing unit where instructions and data are stored.

Total operations: Operations performed only after a group of records has been processed.

Total time: That part of the RPG II progam cycle in which calculation and output operations specified for a group of records are done.

Zero suppression: The elimination of leading zeros in a number. For example, 00057, when zero suppressed, becomes 55557 (5 represents one blank space).

address 2 alphameric constants 102 alphameric field 24 AND relationship 119 arithmetic/logic unit of system 2 arithmetic operation 102

blank-after 59

calculation operations description 40 program cycle operations 40 types of 42 sample job 49 calculation specifications form 41 coding 4 comment lines 133 compare operation 99 compilation 5 compiler 5 constants alphameric 102 definition 42 headings 66 numeric 42, 102 control break 54 control field 54 control level indicators definition 54 program cycle operations 56 RPG II specifications 57 sample job 60 control section of system 2 control specifications form 135

data processing terms 8 decimal positions 20, 47 describing output records 25 desk checking specifications 135 detail output 15 detail records 27 device designation 18 diagnostic messages 139 documentation 133

editing 33 error messages 139 factor 1 42, 99 factor 2 42, 99 field 8 field indicators definition 111 program cycle operations 111 RPG II specifications 112 sample job 114 field location 24, 29 field names 22, 29 files definition 8 input 18 output 18 file description specifications form 16 file designation 18 file names 17, 21, 26 first page (1P) indicator definition 64 program cycle operations 65 RPG II specifications 66 sample job 77

half-adjusting 47 heading records 27

edit codes

34

indicators 53 input 1 input devices 1 input/output operations description 14 program cycle operations 14 sample job 36 input record description 20 input specifications form 20

job 1	36
job 2	49
job 3	60
job 4	77
job 5	89
job 6	105
iob 7	114

last record (LR) indicator definition 74 program cycle operations 75 RPG II specifications 76 sample job 77 LR indicator (see last record indicator) L1-L9 indicators (see control level indicators)

machine language 3

numeric constants 42, 102 numeric field 24

OA-OG, OV indicators (see overflow indicators) object program 3 OR relationship 74, 118 output 1 output devices 1 output record description 25 output specifications form 25 overflow 68 overflow handling 68 overflow indicators definition 68 program cycle operations 70 RPG II specifications 72 sample job 77 overflow line 68 overflow page 68

primary file 18 printed output 1 printed reports 30 printer spacing chart 10 processing unit 1 2 arithmetic/logic control 2 storage 2 program 3 program cycle 11 program cycle operations 12 program listing 137 programming aids 8 programming language 3 programming terms, basic 8 programmer's job 121

record 8 record identification codes 85 record identifying indicators definition 81 program cycle operations 82 **RPG II specifications** 83 sample job 89 record layout form 9 record length 19 record size 19 record type 27 record type sequence 87 reports, printed 30 result field 45 length 47 resulting indicators definition 96 program cycle operations 96 RPG II specifications 98 sample job 105 rounding 47 RPG II program cycle 11 **RPG II programming language** 3 RPG II specification forms 4

skip 72 skipping 31 tapeless carriage control 31 source program compilation 4 description 3 translation 5 spacing after printing 73 description 30 with overflow 72 specifications forms 4 specification form order 134 specifying record identification codes 85 specifying record identifying indicators 86 specifying record type sequence 87 storage section of system 2 system, data processing description 1 input devices 1 output devices 1 processing unit 1

test the program 140 total records 27 type of data 24

1P indicator (see first page indicator) 01-99 indicators (see field indicators, record identifying indicators, resulting indicators) Please use this form only to identify publication errors or request changes to publications. Technical questions about IBM systems, changes in IBM programming support, requests for additional publications, etc, should be directed to your IBM representative or to the IBM branch office nearest your location.

Error in publication (typographical, illustration, and so on). No reply.

Page Number Error

Inaccurate or misleading information in this publication. Please tell us about it by using this postage-paid form. We will correct or clarify the publication, or tell you why a change is not being made, provided you include your name and address.

Page Number Comment

IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Address

Name



4111 Northside Parkway N.W. P.O. Box 2150 Atlanta, Georgia 30301 (U.S.A. only)

General Business Group/International 44 South Broadway White Plains, New York 10601 U.S.A. (International) I

Cut Along Line

Please use this form only to identify publication errors or request changes to publications. Technical questions about IBM systems, changes in IBM programming support, requests for additional publications, etc, should be directed to your IBM representative or to the IBM branch office nearest your location.

Error in publication (typographical, illustration, and so on). No reply.

Page Number Error

Inaccurate or misleading information in this publication. Please tell us about it by using this postage-paid form. We will correct or clarify the publication, or tell you why a change is not being made, provided you include your name and address.

Page Number Comment

No postage necessary if mailed in the U.S.A.

Address

Name

IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may,

of course, continue to use the information you supply.



General Business Group/International 44 South Broadway White Plains, New York 10601 U.S.A. (International)

Cut Along Line

Please use this form only to identify publication errors or request changes to publications. Technical questions about IBM systems, changes in IBM programming support, requests for additional publications, etc, should be directed to your IBM representative or to the IBM branch office nearest your location.

Error in publication (typographical, illustration, and so on). No reply.

Page Number Error

Inaccurate or misleading information in this publication. Please tell us about it by using this postage-paid form. We will correct or clarify the publication, or tell you why a change is not being made, provided you include your name and address.

Page Number Comment

No postage necessary if mailed in the U.S.A.

Address

Name

it believes appropriate without incurring any obligation whatever. You may,

of course, continue to use the information you supply.

IBM may use and distribute any of the information you supply in any way



4111 Northside Parkway N.W. P.O. Box 2150 Atlanta, Georgia 30301 (U.S.A. only)

General Business Group/International 44 South Broadway White Plains, New York 10601 U.S.A. (International) 14-2

I

Cut Along Line

ν.



International Business Machines Corporation

General Systems Division 4111 Northside Parkway N.W. P.O. Box 2150 Atlanta, Georgia 30301 (U.S.A. only)

General Business Group/International 44 South Broadway White Plains, New York 10601 U.S.A. (International)

GC21-7514-2