FORTRA

Robot Reports

OPAL Reference Guide

Copyright Terms and Conditions

202401021008

OPAL Language

The OPerator Assistance Language (OPAL)

Robot Reports provides a customized language for report processing. The language is called OPAL, the OPerator Assistance Language. It is a powerful fourth-generation operations language (4GL) that lets you review the information in a line of report text and take action based on that information.

The OPAL language is easy to learn. It is a fixed format language, like RPG, but its syntax is like CL. To code OPAL statements, you just fill in fields on a panel and Help is available for each field.

If you don't find the information you need in this reference guide, go to the Robot product page on our website. There you can find additional information in our support topics or access contact information needed to connect with a Robot Technical Consultant.

OPAL Code in Message Sets: Before and After

A report set can contain up to four sections of OPAL code:

- Report Segment OPAL
- Exception Distribution OPAL
- Report View OPAL
- Report View OPAL HILITE

See the *Robot Reports Administrators Guide* for more information on each of these features, as well as specific instructions on entering OPAL code for Report Views and Exception Distribution. There are instructions for entering Segment OPAL code later in this manual. Refer to the Examples section of this manual for detailed examples of OPAL code for each of the OPAL types.

Report segment OPAL defines report segments. Report segments are portions of a report containing pages or lines selected from the entire report. Report segment OPAL code defines the rules to be used in selecting those pages or lines in a report segment. In addition, report segment OPAL can be used to incorporate management action thresholds for a report. Such thresholds then can form the basis for highlighting lines in the report or selecting lines or pages to print. Segmentation instructions operate on lines or pages.

Exception Distribution OPAL defines when report distribution should not occur. Normal report distribution is set up to distribute reports to recipients every time the report is processed. If you want distribution only on selected days of the week or month, you must define exception distribution OPAL.

Report view OPAL defines report layouts. A view reformats the report to the needs of the individual report recipient. Report view OPAL is rarely entered by the report administrator. Usually, report recipients create their own views while viewing the report. Recipients can freeze report titles, column headings, and columns, and can move, copy, or exclude columns from their view.

Report view OPAL HILITE allows selected items to be highlighted only when the recipient is using that view to look at the report. This is different from the HILITE command in report segment OPAL, which highlights items within a segment and looks the same to everyone on that report's distribution list.

OPAL Statements

OPAL code consists of a sequence of OPAL statements. The OPAL statements are performed in order unless an IF, WHILE, or GOTO statement changes the processing order.

An OPAL statement is entered on a single line. (OPAL does not use continuation lines.) Each line has several fields as shown in the following example:

```
RBC275                  OPerator Assistance Language                17:34:53
                                                                    LIL
Message Group Name . . : *NONE                          Status: Released
Message Set Name . . . : TWTEST      Testing OPAL code
Monitor for Message ID : CPF0907
Start at Sequence  . . : ___
 Logic
Operand   Variable   Operation    Operation Values                    Seq
 IF       WORKDAY    EQ           Y                                    10
 IF       SYSTIME    GT           170000                               20
 OR       SYSTIME    LT           070000                               30
          ___        *            If workday and after working hours   40
 THEN     ___        DELAY        1800                                 50
          ___        *            If working hours, delay 10 mins      60
 ELSE     ___        ___                                               70
          ___        DELAY        600                                  80
 END      ___        ___                                               90
          ___        *            If working day, vary on workstation 100
          ___        EXECUTE      VRYCFG CFGOBJ(VAR3) CFGTYPE(*DEV...  110
          ___        *            ___                                  120
                                                                    More...
F3=Exit                    F4=Prompt                  F7=ROBOT Variables
F12=Previous               F15=Reply OPAL             F24=More Keys
```

The contents of each field is indicated by its heading. On the IBM i, the last column (Seq) is not part of the OPAL statement; it contains the sequence number of the statement.

These four fields constitute the OPAL statement. Each OPAL example in this reference guide uses a table similar to the one below to show you the fields used by the statement:

Logic Operand	Variable	Operation	Operation Values

OPAL Fields

The following sections describe what you can enter in the OPAL statement fields. The fields are described in the same order that they occur in an OPAL statement: Logic Operands, Variables, Operations, and Operation Values.

The first field in an OPAL statement is the Logic Operand field. Logic operands are used to define logic control for the OPAL code, that is, which OPAL statement is performed next.

Logic Operands

The first field in an OPAL statement is the Logic Operand field. Logic operands are used to define logic control for the OPAL code; that is, which OPAL statement is performed next.

Seven logic control structures are available in OPAL: IF, AND, OR, ELSE, THEN, END, and GOTO. The IF structure specifies conditions that must be met before a set of operations is performed. A GOTO operation changes the next statement processed to the specified TAG statement.

Logic Control

You don't need logic control in your OPAL code if all operations in the code are to be performed for every report processed by that OPAL code. For example, the following report segment OPAL code prints every line of an existing report.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				INCLIN	

Conditional Operations

To make operations conditional, that is, to specify conditions that must be met before other operations are performed, you use Logic Operands in your OPAL code.

The following is an example of an IF-ELSE structure. The operation in the IF structure is performed only if the conditions are met. Program SAL456 is run to update the general ledger if the debits and credits for the day's transactions balance. If they do not, the accountant (Jim) is paged.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF			15 26	EQ	Grand Total
			70 80	CHGTO	USRNBR1
			85 95	CHGTO	USRNBR2
IF	USRNBR1			EQ	USRNBR2
				EXECUTE	CALL SAL456
ELSE					
				PAGE	JIM ('DID NOT BALANCE')

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
END END					

Logic Operand: AND

AND—Add a Condition to a Condition Set

Use an AND statement to add another condition to a set of conditions. (In contrast, an OR statement starts a new condition set.) All conditions in a set must be true for a true result.

You can also use the AND statement to add a condition to an IF condition list. The AND connects the condition to the preceding condition in the list. Both conditions must be true for a true result; if either condition is false, the result is false.

Syntax

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
AND	**[required] or**	**[optional**	**required]**	**required**	**required**

A variable or the columns to check (those in the brackets) must be specified.

Examples

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
AND	USRNBR1			EQ	USRNBR2
AND			88 90	LT	035

Any Number of Conditions in a Set

A condition set can contain any number of conditions. For example, the following IF statement has three conditions (highlighted below) that create a set of conditions that must all be true for the operation to be performed:

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF			12 18	EQ	SKU NO
AND			88 90	LT	035

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
AND			88 90	GT	035
THEN				INCLIN	
END					

Logic Operand: IF

IF—Perform If Conditions are Met

Use an IF statement to test for one or more conditions. If the conditions are true, the operations following the condition list are performed. If the conditions are not true, the program continues at the statement following the operation list.

If the operations should be performed just once if the conditions are true, use an IF statement.

Syntax

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	**[required] or**	**[optional**	**required]**	**required**	**required**

A variable or the columns to check (those in the brackets) must be specified.

Examples

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF			1 30	CT	'Sales Dept'
IF	LINENBR			EQ	1
IF	PAGENBR			LT	LASTPAGE

Multiple Conditions

You can add more conditions to the IF statement by using AND and OR statements after the IF.

An optional THEN statement can mark the end of the condition list and the beginning of the operations to be performed if the conditions are true.

Logic Operand	Variable	Operation	Operation Values
IF	ACTUSR	NE	DAVEJ
AND	SYSTIME	GT	170000
OR	ACTUSR	NE	DAVEJ
AND	WORKDAY	EQ	NO
THEN		PAGE2WAY	OPERATOR
END			

For example, the following statements build a report segment for a product manager that includes only parts that did not achieve the desired margin of 35% or more.

Logic Operand	Variable	Line	Column Beg.	Column End	Operation	Operation Values
IF			12	18	EQ	Part No
AND			88	90	LT	035
					INCLIN	
END						

You may also have multiple conditions that require comparisons between lines on a page of the report.

For example, you may want to create a segment that includes only pages for the payroll department of the lawn sprinkler division. Suppose the division name is on the first line of each page of the report and the department identifier is on the third line. The following OPAL code examines the first three lines of each page of the report. If the condition is true, the page is included, otherwise the page is excluded.

Logic Operand	Variable	Line	Column Beg.	Column End	Operation	Operation Values
IF		1	10	25	EQ	Lawn Sprinklers
AND		3	10	90	EQ	Payroll Dept.
					INCPAG	
					QUITPAGE	
ELSE						
					EXCPAG	
					QUITPAGE	
END						

There are several details worth noticing in this example:

- **Use of the line number field:** If you use a line number with the IF operand, you must use line numbers throughout the entire IF structure including any associated AND or OR statements.
- **Implied IF statement:** This type of structure creates an implied IF statement. The implied statement says, if this is the highest line number referred to in the explicit IF statements, evaluate the IF condition list and take action. With this type of IF structure, the subsequent operation statement is not acted upon until the highest line number needed to evaluate the entire IF condition list is reached.

  In this case, the implied IF statement is located just before the INCPAG operation and the highest line number is 3. The statement says, if this is line 3, evaluate the IF condition list. If the condition is true, include the page; if it is false, exclude the page.
- **Use of the QUITPAGE operation:** This operation is used so that only the first three lines of each page are processed. This makes the segment process more efficiently.

Nested IF Statements

IF statements can be nested. That is, an IF statement can be enclosed in another IF statement.

END Statements Required

For every IF statement, there must be a corresponding END statement. However, END statements can be omitted before ELSE statements and at the end of the program. Consider this program structure:

1	IF	condition list	
2	THEN		operation list
3	END		
4	ELSE		
5	IF	condition list	
6	THEN		operation list
7	END		
8	END		
9			operation list

- The END in line 3 is not necessary because END is not required before ELSE.
- The END in lines 7 is required because this is not the end of the program.
- The END in line 8 is required to end the nested IF structure.
- The 'operation list' in lines 2, 6, and 9 are not part of the nested IF structure.

Logic Operand: OR

OR—Start New Condition Set

Use an OR statement to start a new set of conditions. You can also use the OR statement to start a new set of conditions in an IF condition list. **Note:** To add a condition to an existing set of conditions instead, use the AND statement.

Syntax

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
OR	**[required] or**	**[optional**	**required]**	**required**	**required**

A variable or the columns to check (those in the brackets) must be specified.

Examples

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
OR	PAGENBR			LT	15
OR	STSINC			EQ	YES
OR			12 18	EQ	Part No

OR Starts a New Condition Set

Each OR statement in an IF condition list starts a new set of conditions. For example, the OR statement in the following IF condition list starts a second condition set. Each highlighted set contains one condition; if either condition is true, the result is true.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	TEXT			CT	'Total'
OR			12 18	EQ	Part No
AND			88 90	LT	035
THEN				INCLIN	
END					

Each Condition Set Tested Separately

Each condition set is tested separately. When the same condition is duplicated in all condition sets, it can be pulled out and put into its own IF statement, making a nested IF structure. But, for the condition set to be true, the other condition in the set must also be true.

Logic Operand: THEN

THEN—Start Operation List

The THEN statement is always optional. Use it to mark the beginning of an operation list after an IF or ELSE structure. Only one THEN is allowed for each IF or ELSE.

Syntax

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
THEN	optional	optional	optional	required	optional

Examples

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
THEN				STRINCLIN	
THEN				EXCPAG	

Use in IF Structure

THEN can be used to mark the beginning of each operation list in an IF-ELSE structure as follows:

```
IF          condition list
THEN                        operation list
ELSE
THEN                        operation list
END
```

Logic Operand: ELSE

ELSE—If Conditions Are Not Met

Use the ELSE condition to extend an IF structure so it includes processing to be done only if the preceding IF condition lists are not true.

Syntax

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
ELSE	blank	blank	blank	blank	blank

Example

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
ELSE					

Every ELSE Needs an END

Each ELSE statement must be followed by a corresponding END statement. However, the END can be omitted before another ELSE or at the end of the program.

The simplest IF-THEN-ELSE structure is as follows:

IF	condition list		
THEN		operation list	Operations performed if the condition list is true.
END			
ELSE			
THEN		operation list	Operations performed if the condition list is false.
END			

Extends IF for Additional Conditions

Additional ELSE clauses can be added to process other conditions. Only one operation list in the IF structure is executed. For example:

IF	condition list 1		
THEN		operation list	Done if condition list 1 is true.
END			
ELSE			
IF	condition list 2		
THEN		operation list	Done if condition list 1 is false, but condition list 2 is true.
ELSE			
END			
END			
IF	condition list 3		
THEN		operation list	Done if condition lists 1 and 2 are false, while condition list 3 is true.
END			
END			
THEN		operation list	Done if condition lists 1 and 2 are false, and the first test of condition list 3 is also false.
END			

Logic Operand: END

END—Ends the Operation List

Every IF and ELSE statement must be followed by a corresponding END statement. The END statement marks the end of the processing for the IF or ELSE.

However, an END statement is optional if:

- It is immediately before an ELSE.
- It is at the end of the program.

Syntax

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
END	blank	blank	blank	blank	blank

Example

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
END					

When END is Optional

The following outline shows where END statements are optional:

1	IF	condition list	
2	THEN		operation list
3	END		
4	ELSE		
5	IF	condition list	
6	THEN		operation list
7	END		
8	ELSE		
9	THEN		operation list
10	END		
11	END		

- The END in lines 3 and 7 aren't necessary because END is not required before ELSE.
- The END in lines 10 and 11 are optional because they're at the end of the program.

So, if you remove the optional END operands, the remaining code is as follows:

IF	condition list	

THEN		operation list
ELSE		
IF	condition list	
THEN		operation list
ELSE		
THEN		operation list

OPAL Segments

Accessing Report Segments

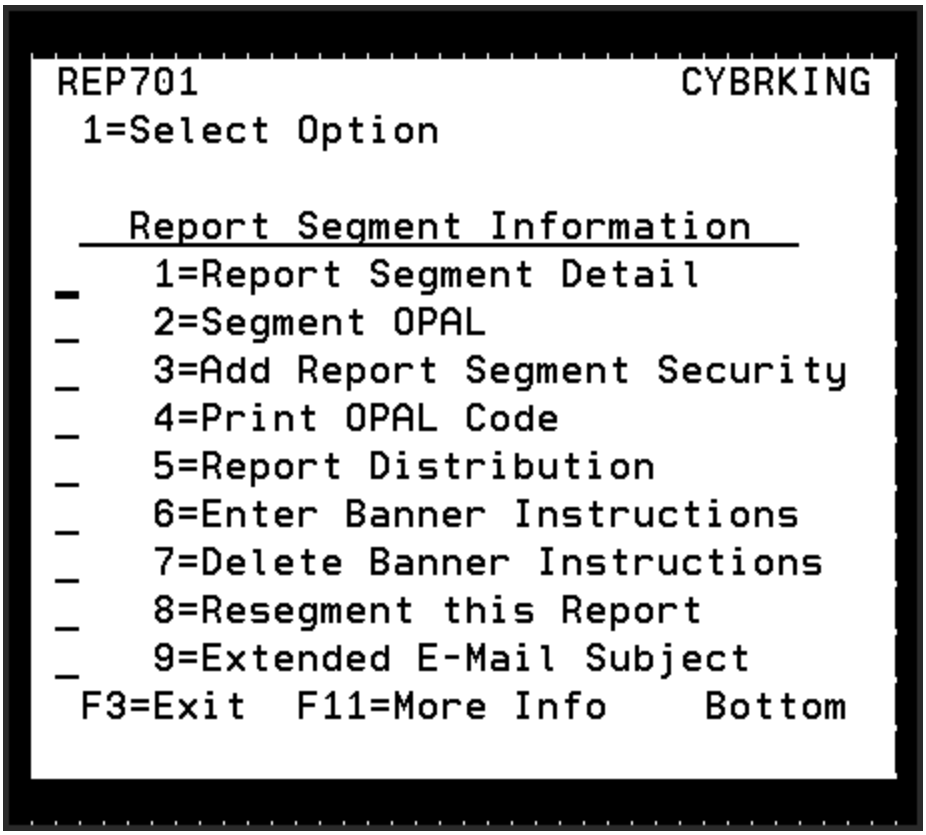
The Maintain Report Segments panel lists the segments defined for a selected report and allows you to maintain report segments.

```
REP224                        Maintain Report Segments                  16:09:14
                                                                        CYBRKING

Report Set . . . . . . . . . : ABLANKLINE test blank line
Report Name  . . . . . . . . : PZC065P1   test blank line
  Start at Report Segment  . : __________
  Search Description for . . : ________________________________
Options
  A=Add/Copy/Delete     1=Select     ?=More Options
                                                        Processing
Opt  Report Segment          Description                  Sequence   OPAL Status
 __    *REPORT          test blank line                       10
 __    BLANK            test blank line                        4    CHANGED




F3=Exit          F4=Prompt             F6=Add Record        F9=Select Search
F12=Previous     F21=System Command    F22=Preceding Opt    F23=Prev Options
                 _
```

Do the following to navigate to the Maintain Report Segments panel:

1. Select option **1**, Report Sets Menu, on the Robot Reports Main Menu.
2. Select option **2** for Maintain Report Sets.
3. Place the cursor next to the report set you want to work with and press **F4** to prompt.
4. Enter a **1** next to Advanced Options.
5. Enter a **1** next to Report Segments in the Advanced Options panel.

From the Maintain Report Segments panel, you can press **F4** next to a report segment to work with OPAL code and other Report Segment options.



Defining Report Segments with OPAL Code

From the Maintain Report Segments panel, select option **2** on the options menu to access the OPAL Report Segment panel. On the OPAL Report Segment panel, enter the OPAL code that defines the report segment.

```
REP240                          OPAL Report Segment              15:57:38
                                                                 CYBRKING
 Report Set . . . . . : ABLANKLINE test blank line
 Report Name  . . . . . : PZC065P1   test blank line
 Report Segment . . . . : BLANK      test blank line
   Start at Sequence  . . : ___
 Logic                  Column
Operand  Variable Line Beg End  Operation   Operation Values                Seq
 IF      __________ ___  1   4  EQ          'Cust'                          10
 IF      __________ ___  26  29 LT          100                             20
 END     __________ ___  ___ ___ __________ ________________________________ 40
 IF      __________ ___  1   4  EQ          'Ship'                          50
         __________ ___  ___ ___ INCLIN     ________________________________ 60
 END     __________ ___  ___ ___ __________ ________________________________ 70
 IF      __________ ___  1   4  EQ          'Cncl'                          80
         __________ ___  ___ ___ INCLIN     ________________________________ 90
         __________ ___  ___ ___ INCNXTLIN  1                               100
 END     __________ ___  ___ ___ __________ ________________________________ 110
         __________ ___  ___ ___ __________ ________________________________ 120
         __________ ___  ___ ___ __________ ________________________________ 1 +

F3=Exit    F4=Prompt     F7=ROBOT Variables    F12=Previous     F14=Services
F18=Resequence     F19=Logic Check     F22=Preceding Opt      F23=Prev Options
```

The OPAL language lets you specify operations to be performed when certain conditions are met. Enter an OPAL statement by filling in one or more fields on a line. The statement fields are as follows:

- **Logic Operand:** Used to set up powerful data conditions with IF-THEN logic.
- **Named Variables:** Specifies the named OPAL variable to be tested or changed.
- **Report Variables:** Specifies the lines or columns to be tested or changed.
- **Operation:** Specifies the operation to be performed or the comparison to be made.
- **Operation Values:** Provides the value to be used by the comparison or operation.

If you create a report segment and don't enter any OPAL code for it, no report will be generated for that segment. However, Robot Reports will place a message in the job log to let you know that an empty segment exists.

You can edit OPAL code at any time. Prompting (**F4**) is provided for logic operands, named OPAL variables, columns, and operations.

Each line of OPAL code has a sequence number. You can resequence the lines by changing the sequence numbers and then pressing **F18**. You can delete a line by blanking out the entire line including the sequence number.

Once you have processed a report through Robot Reports, you can request prompting in the Column field. By pressing **F4**, you will be shown the View a Report panel so you can select the beginning column and ending columns of the field. Simply place your cursor in the beginning column and press **F13**. You will receive a message that you have marked the beginning of the field. Then move your cursor to the ending column and press **F13** again. This procedure is called marking and bounding the field. If you know the numbers of the columns you want to select in advance, you can simply type the numbers without performing the mark and bound procedure.

When you press **Enter** to record your entries, the syntax of each OPAL statement is checked. You can press **F19** to check the logic of the OPAL code. It checks that logic operands are in the correct order and matches TAG and GOTO statements. In addition to using this logic check, you should test your OPAL code against a sample or test report before putting the code into production.

When you are setting up OPAL code for a segment for the first time or when you make changes to your code, you are given the option, when you exit the panel, of creating an OPAL program. If you are not finished with your changes or you do not want your changes used yet, enter an **N** in the window. Your changes to the OPAL code are kept, but the old program is used. Otherwise, enter a **Y** and your OPAL code is created or revised immediately.

Things You Can Do:

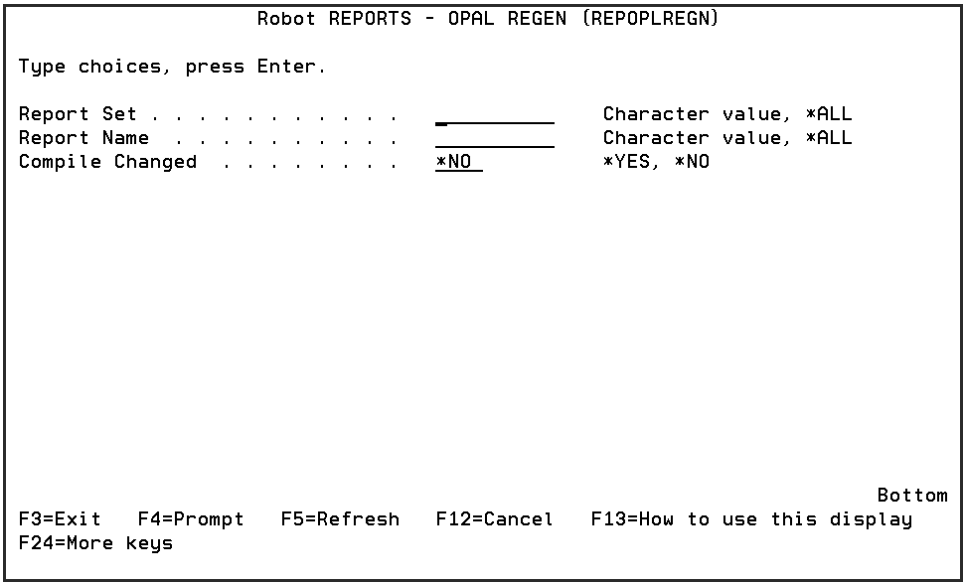
Here are some other things you can do on the OPAL Report Segments panel:

- If you have Robot Schedule installed, press **F7** to display a window listing the Robot Schedule reserved command variables.
- Press **F14** to display the services window that allows you to copy your OPAL code to another segment. In order for the copy to work, the target segment for the copy must already exist, but must not have any OPAL code defined for it. Press **F4** to select from a list of segment names.

Report OPAL Regeneration — REPOPLREGN

The command REPOPLREGN is used when you want to generate and compile a number of OPAL programs at once. You can use *ALL in both the Report Set and Report Name fields to generate and

compile all of the OPAL code that has been changed. This command is designed for Segment OPAL, View Hilite OPAL, and Exception Distribution OPAL. It does not work for Report View OPAL.



Do the following to generate and compile OPAL programs:

1. Enter the name of the Report Set and the Report Name that you wish to generate and compile OPAL for. If you want to generate and compile the OPAL for every Report Set and Report Name, enter ***ALL** in the top two fields.
2. Enter ***YES** if you want to compile all of the code with a changed status. Enter ***NO** if you don't want your changed code compiled.
3. Press **Enter** to submit the command.

Using OPAL Variables

This section explains what should be entered in the Variable field and lists the various OPAL variables that can be used.

Two Types of Variables

There are two types of variables in OPAL for Robot REPORTS – **report variables** and **named variables**. Report variables are defined by the beginning and ending columns where they are found in a line of a report and, optionally, by line number as well. Named variables contain values from the report set being processed. In an OPAL statement you use the variable fields for either of the following:

- To specify the variable to be tested by an IF, OR, or AND statement.
- To specify the variable to be changed by a CHGTO, ADD, SUB, INTABLE, or NOTINTABLE operation.

For all other statements, the field is left blank. The CHGTO, ADD, SUB, INTABLE, or NOTINTABLE operations are described later under Operations.

Conditions

To specify a condition on an IF, AND, or OR statement, you specify three values:

- In the **Variable field**, the OPAL variable with the value to be tested.
- In the **Operation field**, a comparison operator.
- In the **Operation Values field**, an operation value that's either the actual value or an OPAL variable.

When the condition is tested, the values of the Variable and Operation Values fields are compared. If the two values have the relationship specified by the comparison operator, the condition is true. Otherwise, the condition is false.

For example, consider the following IF condition:

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	SYSTIME			GT	170000

SYSTIME is an OPAL variable containing the current system time; the comparison operator is GT; and the operation value is 170000. When the condition is tested, the current system time is compared with the value 170000. If the time is greater than 170000, the condition is true; otherwise, it is false.

Report and Named Variables

This section explains the two types of variables in OPAL for Robot REPORTS – **report variables** and **named variables**.

Report Variables, Line/Column Fields

The line and column fields are report variables that define search areas for a comparison. The more restrictive you are in defining the area of the report to be searched, the faster Robot REPORTS will process the entire report.

The line field is equivalent to a separate IF statement. When both the line and column fields are specified, the statement is processed like an IF statement followed by an AND statement. Whenever possible, you should use the line number together with the column definitions to limit the number of lines Robot REPORTS has to search.

Example: Suppose you have a report that you want to burst electronically. You know that in your report format, the department name occurs on the first line of every page in columns 70 to 79. To determine whether or not to include the page in the report segment, you could have Robot REPORTS check columns 70 to 79 in every line of the report:

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF			70 79	EQ	SALES DEPT
				INCPAG	

This might be necessary if your report doesn't start a new page with each change of department name. But assuming your report does start a new page, you could use this code:

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	TEXT			CT	'Grand Total'
THEN	USRFLD1			CHGTO	TEXT
				EXECUTE	CALL PGM
					PARM(TEXT)

This code will have the same result but can process more quickly.

Named Variables, Variable Field

Named variables contain values that are obtained from the report and report set being processed. In most cases, the variable receives its value when Robot REPORTS begins processing the report. You can use a named variable in the Variable field, in the Operation Value field, or in a command parameter in the Operation Value field.

For example, all of the following statements use the named variable TEXT:

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	TEXT			CT	'Grand Total'
THEN	USRFLD1			CHGTO	TEXT
				EXECUTE	CALL PGM
					PARM(TEXT)

Named Variables

System, Time, and Date

Use the following named OPAL variables to reference the current system, system time, and system date.

SYSTEM — Name of Current System

The **SYSTEM** variable gets the system name when the statement is processed.

Value: System name (up to 10 characters).

Example: The following condition is true if the system name is D10.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	SYSTEM			EQ	D10

SYSTIME — Current Time

The **SYSTIME** variable gets the current time from the system clock when the statement is processed. Use SYSTIME if the report processing should differ depending on the time of day.

Value: Six-digit number representing the time as hours, minutes, and seconds on a 24-hour clock. For example, 123000 is exactly one half-hour after noon.

Example: The first condition is true if the current time is after 5 p.m., but before midnight. The second condition is true if the current time is after midnight, but before 8 a.m.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	SYSTIME			GT	170000
OR	SYSTIME			LT	080000
THEN				PAGE	OPERATOR
END					

SYSDATE — Current Date

The **SYSDATE** variable gets the current system date when the statement is processed. Comparisons using SYSDATE convert the date to yymmdd format.

Value: Six-digit number representing the date in the system format. If the system format is month, day, year (mmddyy), April 15, 2015 is 041515.

Example: The following condition is true if the current date is before July 1, 2015, and the system date format is mmddyy.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	SYSDATE			LT	070115

Robot Schedule Calendar

If you have Robot Schedule installed on your system, you can reference the characteristics of a date as defined by the Robot Schedule calendar. Robot REPORTS uses Robot Schedule's STANDARD calendar. For more information about Robot Schedule calendars, see the Robot Schedule User Guide.

WORKDAY — Is it a Working Day?

The **WORKDAY** variable contains a true value (YES) if today is a working day as defined by the Robot Schedule calendar currently in effect. Otherwise, it contains a NO value.

Value: YES or NO (or Y or N).

Example: The following condition is true if today is not a working day.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	WORKDAY			EQ	NO

DAY — Day of the Week

The **DAY** variable gets the number (1-7) of the current day of the week (Monday, Tuesday, and so on).

Value: Number where Monday is 1, Tuesday is 2, and so forth up to 7 for Sunday.

Example: The following condition is true if today is a Friday.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	DAY			EQ	5

DAYMTH — Day Number in the Month

The **DAYMTH** variable contains the day number of today within the current month as defined by the Robot Schedule calendar currently in effect.

Value: One- or two-digit day number counting from the beginning of the month. The month-ends are defined in the Robot Schedule calendar used. For example, if a fiscal month ends August 28, the day number for August 29 is 1.

Example: The following condition is true if today is after the tenth day of the month.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	DAYMTH			GT	10

LASTDAY — Last Date in the Month

The **LASTDAY** variable contains the date of the last day of the current month as defined by the Robot Schedule calendar currently in effect.

Value: Six-digit date.

Note: If you compare the LASTDAY variable with a date in a Robot Schedule reserved command variable, make sure that the date in the Robot Schedule variable is in yymmdd (year, month, day) format.

Example: The following condition is true if today is the last day of the month.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	LASTDAY			EQ	SYSDATE

WEEKNO — Week Number

The **WEEKNO** variable contains the week number of today within the current month. The first seven days of the month are week 1, the next seven days are week 2, and so forth. The previous month end is defined by the Robot Schedule calendar. For example, if a fiscal month ends August 28, the week number for August 29 is 1.

Value: One-digit number where 1 is the first week in the month, 2 is the second, and so forth.

Example: The following condition is true if today is after the first week of the month.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	WEEKNO			GT	1

Spool File Attributes

The following named OPAL variables contain information about the spool file that Robot REPORTS is using as the basis for its report formatting functions (segmentation and highlighting). The variables contain the identification of the job (by name, number, user, and user data), the spool file number, the output queue and library that the report is on, and the type of forms that should be loaded in the printer.

JOB — Job Name

The **JOB** variable contains the job name.

Value: Job name (up to 10 characters).

Example: The following condition is true if the job name is AR.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	JOB			EQ	AR

&JOBNUMBER — Job Number

The **&JOBNUMBER** variable contains the job number. Use this value as the parameter for a command or to pass the value to another program.

Value: Job number (6 numeric characters).

Example: The following statement displays job information.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				EXECUTE	DSPJOB JOB (&JOBNUMBER/USER/JOB)

USER — Job User

The **USER** variable contains the name of the user profile that ran the job.

Value: User profile name (up to 10 characters).

Example: The following condition is true if the job user is Johnson.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USER			EQ	JOHNSON

USRDTA — User Data

The **USRDTA** variable contains user data on the spool file.

Value: User data (up to 10 characters).

Example: The following condition is true if the user data on the spool file equals ACT125.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRDTA			EQ	ACT125

OUTQ — Output Queue

The **OUTQ** variable contains the name of the output queue from which the report was retrieved.

Value: Output queue name (10 characters).

Example: The following condition is true if the output queue is named BDOUTQ.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	OUTQ			EQ	BDOUTQ

OUTQLIB — OUTQ Library

The **OUTQLIB** variable contains the name of the library in which to find the OUTQ.

Value: Output queue library name (up to 10 characters).

Example: The following condition is true if the output queue is in QGPL.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	OUTQLIB			EQ	QGPL

SPLNAME — Spool File Name

The **SPLNAME** variable contains the name of the spool file that Robot REPORTS is working on.

Value: Spool file name (up to 10 characters).

Example: The following condition is true if the spool file name is equal to REGSALES.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	SPLNAME			EQ	REGSALES

SPLNBR — Spool File Number

The **SPLNBR** variable contains the spool file number of the spool file Robot REPORTS is working on.

Value: Spool file number (up to 10 characters).

Example: The following condition is true if the spool file number is equal to 000001.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	SPLNBR			EQ	000001

FORMTYPE — Type of Form

The **FORMTYPE** variable contains information about the type of forms the printer should be loaded with for this report.

Value: Type of form (up to 10 characters).

Example: The following condition is true if standard forms (plain paper) are to be used.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	FORMTYPE			EQ	*STD

Report Attributes

Various report attributes can be referenced by using the following OPAL variables. These attributes include position in the report by line or page and the ability to look for text references. These named OPAL variables apply only to lines containing text.

FIRSTLINE — First Line of Report

The **FIRSTLINE** variable answers the question, "Is this the first line of the report?" Use this variable to set the condition for your beginning of report procedures. Put it at the beginning of your OPAL code.

Value: YES or NO (or Y or N).

Example: The following condition is true if this is the first line of the report.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	FIRSTLINE			EQ	Y

LASTLINE — Last Line of Report

The **LASTLINE** variable answers the question, "Is this the last line of the report?" Use this variable to set the condition for your end of report procedures. Put it at the end of your OPAL code.

Value: YES or NO (or Y or N).

Example: The following condition is true if this is the last line of the report.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	LASTLINE			EQ	Y

LINENBR — Current Line Number

The **LINENBR** variable contains the number of the line on the page currently being processed by the report segment. LINENBR is always a number between one and the number of lines per page. Use this variable to limit the area of the page Robot REPORTS is searching for columns.

Value: Numeric in the format (4, 0).

Example: If you know the columnar information on the report will always be between lines 8 and 60, you should use this named OPAL variable to speed up the search.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	LINENBR			GT	8
AND	LINENBR			LT	60

Tip: To get accurate line numbers, display the IBM spool file. Enter **+1** in the Control Field to move through the spool file line by line until you reach the line you want to reference in your OPAL code. Use the line number shown at the top right corner of the display as your operation value.

PAGENBR — Current Page Number

The **PAGENBR** variable contains the current page number of the report.

Value: Numeric in the format (9, 0).

Example: The following condition is true if this is page 15.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	PAGENBR			EQ	15

Tip: To get accurate page numbers, display the IBM spool file. Enter **P+** in the Control Field to move through the spool file page by page until you reach the page you want to reference in your OPAL code. Use the page number shown at the top right corner of the display as your operation value.

LASTPAGE — Number of Last Page

The **LASTPAGE** variable contains the number of the last page of the report.

Value: Numeric in the format (9, 0).

Example: The following condition is true if this is the last page of the report.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	LASTPAGE			EQ	PAGENBR

TEXT — Report Text

The **TEXT** variable contains an entire line of the report. Use this variable when you need to pass a text line to another program or to a message.

Value: Up to 378 characters in the text string in the same case (upper or lower) as they appear in the report.

Example: The following OPAL statement passes a text line to program SAL256.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				EXECUTE	CALL SAL256 PARM(TEXT)

STSINC — Status Include

The **STSINC** variable answers the question, “Is this line included in this report segment?”

Value: YES or NO (or Y or N).

Example: The following condition is true if the line being processed is included in the report segment.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	STSINC			EQ	Y

REPORTSET — Report Set Name

The **REPORTSET** variable contains the name of the report set currently being used.

Value: Report set name (10 characters)

Example: The following condition is true if the report set name is equal to AMTRANCYCL.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	REPORTSET			EQ	AMTRANCYCL

REPORTNAME — Report Name

The **REPORTNAME** variable contains the report name currently being used.

Value: Report name (10 characters)

Example: The following condition is true if the report name is equal to ACCT1422.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	REPORTNAME			EQ	ACCT1422

SEGMENT — Report Segment Name

The **SEGMENT** variable contains the name of the report segment currently being used.

Value: Report segment name (10 characters)

Example: The following condition is true if the report segment name is equal to EASTERN.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	SEGMENT			EQ	EASTERN

RECIPIENT — Report Recipient Name

The **RECIPIENT** variable contains the name of the report recipient currently being processed. This variable is only valid for Exception Distribution OPAL.

Value: Report recipient name (35 characters)

Example: The following condition is true if the report recipient name is equal to ANNA.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	RECIPIENT			EQ	ANNA

RCPPRF — Recipient User Profile

The **RCPPRF** variable contains the user profile associated with the current report recipient. This variable is only valid for Exception Distribution OPAL.

Value: Report recipient user profile (10 characters)

Example: The following condition is true if the report recipient user profile is equal to ANNA.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	RCPPRF			EQ	ANNA

DISTTYPE — Distribution Type

The **DISTTYPE** variable answers the question, “What type of distribution is being processed?” This variable is only valid for Exception Distribution OPAL.

Value: P (print only distribution) or B (both view and print distribution)

Example: The following condition is true if the report distribution is print only type distribution.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	DISTTYPE			EQ	P

BUNDLE — Bundle Code

The **BUNDLE** variable answers the question, “Is the report being bundled?” This variable is only valid for Exception Distribution OPAL.

Value: YES or NO (or Y or N)

Example: The following condition is true if the report is being bundled.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	BUNDLE			EQ	Y

User Variables

OPAL also provides named user variables that contain no value until you assign one. You can assign a value to a user variable using the CHGTO operation and later reference the variable in a condition or operation. After a user variable has a value assigned, it can be compared to constants or other variables. When you use a user variable in the Operation Value field, always enter the variable name in uppercase format.

USRFLDn — Character Variables

The five **USRFLDn** variables can contain up to ten characters each, and are referenced as USRFLD1, USRFLD2, ...up to USRFLD5.

Value: Up to ten characters.

Example: The following condition is true if USRFLD1 contains the characters TOTAL.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRFLD1			EQ	TOTAL

USRLONGn — Longer Character Variables

The five **USRLONGn** variables are like the USRFLDn variables except they can handle character strings up to 45 characters each. They are referenced as USRLONG1, USRLONG2, ...up to USRLONG5. They are especially useful with the concatenation operations CAT and BCAT.

Value: Up to 45 characters.

Example: The following condition is true if USRLONG1 contains the characters Department 23.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRLONG1			EQ	Department 23

USRFLGn — Flag Variables

The five **USRFLGn** variables can contain one character each. They are referenced as USRFLG1, USRFLG2, ...up to USRFLG5.

Value: One character.

Example: The following condition is true if USRFLG1 contains the character Y.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRFLG1			EQ	Y

USRNBRn — Floating Point Variables

The five **USRNBRn** variables can contain a number in the format (15,5). They are referenced as USRNBR1, USRNBR2, ...up to USRNBR5.

Value: Numeric in format (15,5).

Example: The following is true if USRNBR1 is greater than 9999.99.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRNBR1			GT	9999.99

USRPAGNn — Page Number Variables

The five **USRPAGNn** variables can contain a number up to nine digits in size (999999999) and are used to check for the page number of a report. They are referenced as USRPAGN1, USRPAGN2, ...up to USRPAGN5.

Value: Numeric string from 1 to 999999999.

Example: The following condition is true if the current page number is greater-than-or-equal-to 1000 (USRPAGN1) and less-than-or-equal to 1200 (USRPAGN2).

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRPAGN1			GE	1000
AND	USRPAGN2			LE	1200

USRCNTn — Numeric Variables

The five **USRCNTn** variables contain a number in the format (5,0). They are referenced as USRCNT1, USRCNT2, ...up to USRCNT5.

Value: Numeric in format (5,0).

Example: The following condition is true if USRCNT1 is less than 7.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRCNT1			LT	7

Example: The following condition is true if USRCNT1 is greater than USRCNT2.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRCNT1			GT	USRCNT2

Operations

The purpose of the OPAL code in a message set is to tell Robot Reports what to do to process a spool file. The OPAL statements that tell Robot Reports what to do are called operation statements.

Operation statements have three purposes:

- To create report segments by selecting certain lines or pages.
- To automate operator procedures that have to do with reacting to information in the report.
- To highlight report information that needs management action.

OPAL code must include at least one operation statement. Any other statements in the OPAL code just determine which operation statements are performed.

The general syntax of an operation statement is as follows:

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
optional	optional	optional	optional	required	optional

When you code an operation statement, you always fill in the Operation field; you provide an Operation Value if one is needed by the operation. Usually, the Logic Operand field is left blank, unless the operation is part of a THEN statement.

For example, the following statement uses only the Operation field.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				INCLIN	

Most operation statements use the Operation and Operation Value fields, as follows:

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				EXECUTE	CALL PGM1

The following CHGTO operation sets the variable USRFLD1 to the value found in columns 10 to 15:

Logic Operand	Variable	Line	Column Beg. End		Operation	Operation Values
			10	15	CHGTO	USRFLD1

Comparisons

When you specify a condition, you must specify a comparison in the Operation field. The comparison specifies a relationship between the value of the OPAL variable in the Variable field and the value in the

Operation Value field. If the relationship exists, the condition is true.

Example: The following condition specifies the greater than comparison (GT) in the Operation field. For the condition to be true, the system time (specified by the OPAL variable SYSTIME) must be greater than the operation value (190000).

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	SYSTIME			GT	190000

The following sections describe the comparisons that are available in OPAL.

Equality Comparisons

For equality, the comparisons you can specify in the Operation field are as follows:

EQ	Equals; the values match exactly.
NE	Not equals; the values do not match exactly.

Order Comparisons

The following comparisons are for comparing the order of the values. For numeric values, a numeric comparison is done. For alphanumeric values, an alphanumeric comparison is done, character by character, from left to right. For example, the numeric value 12 is greater than both 1 and 2. But, the alphanumeric value 12 is greater than 1, but less than 2.

Column values also can be compared. For example, if you are comparing columns 14 and 15 to the value 12, the value in the columns will be converted to a numeric value.

GT	Greater than
GE	Greater than or equal to
LT	Less than
LE	Less than or equal to

Containing Comparisons

The following two comparisons compare the characters to see if they contain or don't contain the values you're looking for. These comparisons are case-sensitive. When it tests the condition, OPAL looks for the same sequence of characters from the operation value in the OPAL variable value. For example, the character sequence SALE is contained in SALES, NEWSALE, and OLDSALES, but it's not contained in SALUPD.

CT	Contains; the character sequence from the operation value is also in the OPAL variable value.
DC	Doesn't contain; the character sequence from the operation value is not in the OPAL variable value.

OPAL Table Comparisons

These two comparisons are used with OPAL tables. The data in the variable field is compared to all elements in the specified OPAL table. The match must be exact.

INTABLE	The variable data exists in the OPAL table.
NOTINTABLE	The variable data does not exist in the OPAL table.

Example: You have a report about the productivity of sales representatives. It generates a page about each rep. The Regional Sales Manager wants only the pages for the representatives in her region. In the past, the operator sorted the pages manually. Now, Robot Reports can automate this task. First you have to create an OPAL table for the Regional Sales Manager; let's call it JONESREPS. Then you enter each of her sales representatives' names, **exactly as they appear in the report**, as elements in that table. Finally, you can create a report segment for her with the following OPAL code:

Logic Operand	Variable	Line	Column Beg.	Column End	Operation	Operation Values
IF		4	75	105	INTABLE	JONESREPS
THEN					INCPAG	

Creating Report Segments

A major reason for using OPAL code is so that you can create report segments. A report segment is the lines or pages selected by the code to be printed or viewed separately from the entire report. A report segment can be:

- An entire report from one spool file
- A few pages from a large report selected based on data on the page
- A few lines from a large report selected based on data in the line
- A report section (for example, departmental reports)
- Any combination of the above

This means you can segment a very large report to give users only the report pages or lines that pertain to their jobs.

Example: Suppose you have an overhead expense report that includes all departments. You want to build a report segment for the sales department that includes only the pages with “SALES DEPT” printed on them. This is very easy to code in OPAL.

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
IF		03	70	79	EQ	SALES DEPT
					INCPAG	
END						

OPAL’s segment building instructions are coded in the operation field.

Exception: An optional THEN is permitted in the Logic Operand field. Using THEN in this field may make your code more readable for you.

Special Segments

When you define a report name for a report set, Robot Reports generates a report segment that contains the entire report called ***REPORT**. If you don't want to create other segments, simply leave it alone. You don't have to enter any OPAL code to use ***REPORT**.

However, if you want to create report segments other than ***REPORT**, you may want to create the special segment ***REMAINDER**, too. ***REMAINDER** accumulates everything that isn't included in the other segments you define. It functions like a scrap bucket and is useful for auditing your work.

Line Operations

The following operations are used to include or exclude lines. For these operations, the operation field is the only field used on the line.

INCLIN– Include this line in the report segment.

EXCLIN– Exclude this line from the report segment.

STRINCLIN– Start including lines here. Include this line and all the lines that follow until the end of the report is reached or another segmenting instruction is received.

ENDINCLIN– Stop (end) including lines with this line. Include this line then exclude the lines that follow until the end of the report is reached or another segmenting instruction is received.

STREXCLIN– Start excluding lines here. Exclude this line and all the lines that follow until the end of the report is reached or another segmenting instruction is received.

ENDEXCLIN– Stop (end) excluding lines here. Include this line and all the lines that follow until the end of the report is reached or another segmenting instruction is received.

The Effect of Excluding Report Lines

When a line is excluded from a report, Robot Reports normally will print a blank line. Robot Reports does this so that it will not disrupt the existing pagination of the report. However, if all the lines on a

page are excluded, the page will not be printed.

There is an alternative if you are selecting a few lines from a large report. When you are defining the report segment on the Report Segment panel, you can choose not to retain pagination. Robot Reports then creates a brand new report containing only the selected lines with new pagination. It retains heading lines.

Page Operations

The following operations are used to include or exclude pages. For these operations, the operation field is the only field used on the line.

INCPAG – Include this page in the report segment.

EXCPAG – Exclude this page from the report segment.

STRINCPAG – Start including pages here. Include this page and all the pages that follow until the end of the report is reached or another segmenting instruction is received.

ENDINCPAG – Stop (end) including pages here. The current page is included; the next page and all the pages that follow are not included until the end of the report is reached or another segmenting instruction is received.

STREXCPAG – Start excluding pages here. Exclude this page and all the pages that follow until the end of the report is reached or another segmenting instruction is received.

ENDEXCPAG – Stop (end) excluding pages here. Include this page and all the pages that follow until the end of the report is reached or another segmenting instruction is received.

Note: If you use Robot Reports to repaginate the report on the Report Segment panel, you cannot use OPAL page operations to build your segment. Page operations do not apply to segments where you have selected Retain Paging = N.

Selection Mode

Another factor that affects how your OPAL code is processed is the selection mode in Robot Reports. When Robot Reports is in include mode, it assumes it should include everything you haven't specifically excluded. Exclude mode is just the opposite.

The following table shows you the mode in which Robot Reports will start processing.

OPAL statements defining segment	Robot Reports starts processing in...	Notes
OPAL statements use only include operations: INCLIN, INCPAG, STRINCLIN, STRINCPAG, INCNXTLIN, and INCPRVLIN	EXCLUDE mode. Robot Reports assumes it should exclude anything you have not explicitly included.	Include operations always contain the string INC.
OPAL statements use only exclude operations: EXCLIN, EXCPAG, STREXCLIN, STREXCPAG, EXCNXTLIN, EXCPRVLIN	INCLUDE mode. Robot Reports assumes it should include anything you have not explicitly excluded.	Exclude operations always contain the string EXC.
OPAL statements use both include and exclude operations	EXCLUDE mode. When both include and exclude operations are present, Robot Reports defaults to EXCLUDE mode.	These statements can contain any include or exclude operation. You must be careful when you are developing these OPAL statements to get the correct segments.
There are no OPAL statements, so operations are not applicable.	INCLUDE mode. Becuase there are no OPAL statements, Robot Reports assumes you want to include everything.	A segment with no OPAL statements is treated similarly to the *REPORT segment.

How Selection Mode Changes

As Robot Reports processes statements, its mode changes.

Example: Suppose you want to create a segment of a sales report for the Eastern regional sales manager. He wants all the records from his own region. In addition, he wants to know about any sales over $10,000 in other regions.

You can check what selection mode you are in by using the status include (STSINC) variable. If STSINC equals yes, you are in include mode. If STSINC equals no, you are in exclude mode.

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
IF		1	1	10	EQ	EASTERN
					INCPAG	
END					EQ	NO
IF	STSINC				GT	10000
AND			60	70	INCLIN	
END						

In this case, the OPAL code uses only include operations so Robot Reports begins processing in exclude mode. Robot Reports shifts to include mode when it processes the first line of a page that has EASTERN in columns 1 to 10 and continues in include mode until the first page that does not have EASTERN in columns 1 to 10. It then switches back to exclude mode. It will exclude the remainder of the report unless it finds a sales record in columns 60 to 70 over 10000. If it finds such a record, it changes to include mode for that line, then returns to exclude mode.

Line Block Operations

Robot Reports also provides line block operations. Line block operations allow you to work with records in a report that occur on multiple consecutive lines. Line block operations can be used with both fixed and variable length blocks.

INCNXTLIN — Include Next X Lines

Use the **INCNXTLIN** operation to include the next x lines in a report segment.

Operation Value: The number of lines to include. This number can be a constant or a variable. The value of x cannot be greater than the number of lines between the current line and the last line of the page. For instance, if your report has pages that are 66 lines long and you are on line 60, x would have to be less than or equal to six.

Example: Your report includes information in two-line blocks about each part number your company manufactures. You want to create a segment that will pick up selected blocks. First you would create an OPAL table listing the part numbers to look for. Then the following code would pick up the blocks.

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
IF			10	15	INTABLE	PARTS
					INCLIN	
					INCNXTLIN	1
END						

INCPRVLIN — Include Previous X Lines

Use the **INCPRVLIN** operation to include the previous x lines.

Operation Value: The number of lines to include. This number can be a constant or a variable. The value of x cannot be greater than the number of lines between the current line and the first line of the page.

Example: Your report includes information in line blocks of varying length. Each line block is separated from the next block by a blank line.

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
IF			1	25	NE	BLANK
	USRCNT1				CHGTO	1
END						
	USRCANT1				ADD	1
IF			1	25	EQ	BLANK
					INCPRVLIN	USRCNT1
END						

EXCNXTLIN — Exclude Next X Lines

Use the **EXCPRVLIN** operation to exclude the next x lines.

Operation Value: The number of lines to exclude. This number can be a constant or a variable. However, this number cannot be greater than the number of lines between the current line and the last line of the page.

Example: Your report has information about the parts your company manufactures in four-line blocks. You want to create a segment that excludes information about selected parts. List the parts in an OPAL table and then use this code to exclude the selected parts from the segment.

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
IF			1	15	INTABLE	PARTS
					EXCLIN	
					EXCNXTLIN	3

EXCPRVLIN — Exclude Previous X Lines

Use the **EXCPRVLIN** operation to exclude the previous x lines.

Operation Value: The number of lines to exclude. This number can be a constant or a variable. However, this number cannot be greater than the number of lines between the current line and the first line of the page.

Example: Your report includes customer names and addresses in four-line blocks. You want to exclude all the customers in New Mexico from your segment. The state abbreviation is in a field on the fourth line. Therefore, you won't know whether or not to exclude the record until you process the fourth line of the record.

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
IF			22	23	EQ	NM
					EXCLIN	
					EXCPRVLIN	3
END						

Using QUIT, QUITPAGE, and EXCREPORT to Speed Processing

You may have a segment that consists of lines found only at the beginning of a report. For example, you might have a segment for Department 001. All Department 001 records are listed first in the report. In this case, you would want to use a QUIT operation as soon as the department number changes. QUIT breaks the connection between the segment and Robot Reports. Robot Reports will not process this segment any more after a QUIT operation is performed for the segment. By using the QUIT operation well, you can save a lot of processing time.

QUITPAGE stops processing temporarily – only for the length of a page. Its effect on processing time may be less dramatic than using QUIT, but it is very useful when you want to pick out several pages for a segment that are scattered throughout a report.

If there are situations where you may exclude the report entirely, it can be good to put the EXCREPORT operation at the beginning of your report segment OPAL. EXCREPORT removes the report segment and nothing will be distributed for that segment.

QUIT — Stop All Processing

When using OPAL processing type 1, normal processing, the **QUIT** operation stops all processing of the report segment and says that the segment is ready for distribution. When using OPAL processing types 2 or 3, the QUIT operation will stop processing the current segment and start processing the next segment. You should use this operation whenever possible to speed processing. If it is possible to test for information in the report that would mean no more pages should be included in the segment, code the test and use the QUIT operation.

Operation value: None.

Example: The following statement stops all processing of the segment.

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
					QUIT	

QUITPAGE — Stop Processing This Page

The **QUITPAGE** operation stops processing of the report segment until the next page is reached.

Note: QUITPAGE is a page operation. Page operations do not apply to segments where you have selected Retain Paging = N on the Report Segment panel.

Operation value: None.

Example: You want to create a segment for department 15 that will process as fast as possible. If the page does not say “Dept. 15” in columns 15 - 25 of line 2, you want to exclude the page from the segment and skip processing the rest of the lines on that page. The following statements will create your segment.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	LINENBR			EQ	2
IF			15 25	NE	Dept. 15
				EXCPAG	
ELSE					
				INCPAG	
END					
				QUITPAGE	
END					

EXCREPORT — Exclude Report

The **EXCREPORT** operation removes the segment and nothing will be distributed from the segment. This operation may reduce the number of report sets you need. It may also be useful if you work with spool files, such as those downloaded from a mainframe, that have the same identifying information (spool file attributes).

Operation value: None.

Example: You have a job that runs seven days a week, but you only want Robot Reports to process and distribute the report on workdays.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	WORKDAY			EQ	NO
				EXCREPORT	

Example: You can also use this operation to work with spool files that have the same identification. In other words, you need to start to process the spool file to know if it is one you want Robot Reports to process. In this case, you would test for the presence or absence of an identifying feature early in the report, perhaps a title in the first or second line.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	LINENBR			EQ	1
AND	TEXT			DC	'Sales Report'
				EXCREPORT	

Indexing

OPAL gives you the ability to create simple indexes. You can use this capability to make retrieving individual documents from your spool files faster. An index spans all the runs of a report and is thus an access point to the entire Robot Reports archive. For example, you may a have spool file of daily invoices. Each invoice has its unique number and occupies its own page. Therefore, an index of invoice numbers would work well.

The index function works best when it is used to retrieve information that occurs only once per page. Adding an index increases report processing time and storage requirements. Therefore, create indexes to retrieve only frequently used documents.

Note: Indexing can be used only with OPAL processing type 1 (shared).

INDEX — Index a Column Range or Variable Name

The **INDEX** operation stores index values with an archived report so that indexed information can be retrieved very quickly. The INDEX operation can store the value of a variable or column range. The maximum length for an index entry is 45 characters.

Operation Value: Name of the index.

Example: Your sales representatives and customer service personnel often need to retrieve sales invoices quickly while they are on the phone with customers. The index value they need is the sales invoice number.

Creating an index is easy. First, go to the Maintain Report names panel and select option **12** to set up the index. In this case, you might name the sales invoice index SALINV. You have to name and define the index before you can use it as an operation value in your OPAL code. Then you would add a couple lines to your OPAL code. To put the invoice number that appears on line 8 in columns 1 to 10 of each page of your report into the report's index, use the following OPAL code:

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	LINENBR			EQ	8
			1 10	INDEX	SALINV

You can also use variables like USRFLD1 to store index values. For example, if your document has multi-line records and columns 1-10 are blank on many lines, you could index only the lines with non-blank values by using the following OPAL code:

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
			1	10	CHGTO	USRFLD1
IF	USRFLD1				NE	BLANK
	USRFLD1				INDEX	SALINV

Concatenation

OPAL gives you the ability to concatenate information from several fields into a single variable.

CAT — Concatenate Information

The **CAT** operation concatenates specified characters.

Operation value: USRFLDn or USRLONGn.

Example: The following OPAL code concatenates the customer number from columns 1 to 5 and the part number from columns 16 to 19 and then stores the resulting character string in the index named CUSTOMER.

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
	USRLONG1				CHGTO	BLANK
			1	5	CAT	USRLONG1
			16	19	CAT	USRLONG1
	USRLONG1				INDEX	CUSTOMER

BCAT — Insert a Blank and Concatenate

The **BCAT** operation works the same way as CAT but inserts a blank between the items it is concatenating.

Operation value: USRFLDn or USRLONGn.

Example: You might want your index to reflect customer number followed by a space, customer name followed by a space, and order date. The following OPAL code concatenates the customer number from columns 1 to 12, the customer name from columns 25 to 49, the order date from columns 75 to 80, and then stores the resulting character string in the index named CUSTOMER.

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
	USRLONG1				CHGTO	BLANK
			1	12	CAT	USRLONG1
			25	49	BCAT	USRLONG1
			75	80	BCAT	USRLONG1
	USRLONG1				INDEX	CUSTOMER

The entry that is placed in the index as a result of this code could look like this:

540237000065 Smith Landscaping 091594

The ability to insert blanks when you are concatenating information can make the resulting index entry easier to read.

Highlighting

OPAL gives you the ability to highlight information in a report segment or a report view. You can use this capability to make report viewing both more pleasurable and more efficient. For instance, report users may find it easier to scan for information if you put headings in one color, totals in another, and the body of the report segment in a third color. In addition, if you are creating a report segment for management, you can incorporate action thresholds to automatically highlight items requiring management attention.

HILITE — Highlight a Line

The **HILITE** operation highlights a line (or lines) of a report with the color specified in the operation value field. There are 26 colors available. If your users don't have color terminals or PCs, you can still select among several highlight options such as reverse image and underscore. On printed versions of the report segment, highlighted items are double-struck or bolded by the printer.

Operation value: A color selected from the color finder. Press **F4** in the operation value field of the OPAL entry panel to access the finder.

Example: The following OPAL code will display a report segment with the first line of each page in blue and the last line in pink.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	LINENBR			EQ	1
				HILITE	BLUE
END					
IF	LINENBR			EQ	60
				HILITE	PINK
END					

Example: You want to highlight significant past-due accounts. You have an aged trial balance report with 60-day-old amounts in columns 70 to 80 and 90-day-old amounts in columns 85 to 95. You want amounts over $15,000 and 60 days old displayed in yellow and amounts over $10,000 and 90 days old displayed in red. Here's how to code these highlighting rules:

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF			70 80	GE	15000
THEN				HILITE	YELLOW
END					
IF			85 95	GE	10000
THEN				HILITE	RED

Change OPAL Variable Values

These operations are used to perform arithmetic on report columns so that you can automate your report balancing procedures. Robot Reports will remove all the edit codes from a report column whenever you tell it to do an arithmetic operation or to compare the value of a report column to a numeric field or constant. By coding OPAL statements, you can add up report columns, compare totals, and compute differences. If you have Robot Schedule installed, you can use Robot Schedule reserved command variables (for more information, see the section on reserved command variables) to compare totals from different reports.

You also can use these operations to change the values of OPAL variables, especially the user variables. The user variables can be used to control program logic or to hold operation values to be used later.

These operations are described in the following sections.

CHGTO—Change the Value of a User Variable

Use the **CHGTO** operation to assign a value to a user variable. Enter the name of the user variable (USRFLD*n*, USRFLG*n*, or USRNBR*n*, where *n* is 1 to 5) in the Variable field.

Operation value: The value to be assigned to the user variable. For USRFLD*n*, specify up to ten characters; for USRFLG*n*, specify one character; for USRNBR*n*, specify a numeric value (for example: 5,0).

Example: The following statement changes the USRFLD1 value to DAVEJ.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRFLD1			CHGTO	DAVEJ

Example: The following statement changes the value of USRFLD1 to the value found in columns 70 to 79.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
			70 79	CHGTO	USRFLD1

ADD—Add to a Numeric User Variable

Use the **ADD** operation to add to the numeric user variable specified in the Variable field (USRNBR*n*, where *n* is 1 to 5).

Operation value: The number to be added to the numeric value already in the USRNBR*n* variable. For USRNBR*n*, specify a numeric value in the format (15, 5); for USRCNT*n* specify a numeric value in the format (5, 0). The number to be added can also be specified by a USRNBR*n* variable.

Example: The following statement adds 1 to the value in the USRNBR1 field.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRNBR1			ADD	1

Example: The following statement adds the value in the USRNBR1 variable to USRNBR2.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRNBR2			ADD	USRNBR1

Example: The following statement adds the value in columns 30 to 35 to the value in USRNBR1.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
			30 35	ADD	USRNBR1

Example: The following statement adds the value found in a Robot Schedule reserved command variable to the value in USRNBR1. This works only when the Robot Schedule reserved command variable returns a numeric value.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
	USRNBR1			ADD	@@RBTNBR

SUB—Subtract from a Numeric User Variable

Use the **SUB** operation to subtract from the numeric user variable specified in the Variable field (USRNBR*n*, where *n* is 1 to 5).

Operation value: The number to be subtracted from the numeric value already in the USRNBR*n* variable. The number to be subtracted can also be specified by a USRNBR*n* variable.

Example: The following statement subtracts 1 from the value in the USRNBR1 variable.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
	USRNBR1			SUB	1

Example: The following statement subtracts the value in the USRNBR1 variable from USRNBR2.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRNBR2			SUB	USRNBR1

Operations: DELAY — Delays Processing

The DELAY operation has Robot Reports wait a specified number of seconds before it processes the next statement. For example, a delay might be needed to allow an earlier command to finish processing.

Operation value: Number of seconds Robot Reports should wait.

Example: The following statement delays processing for 60 seconds.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				DELAY	60

Execute a Command

Use this operation to execute any IBM i command. See the following section for details.

EXECUTE — Execute any Command

The **EXECUTE** operation can execute any IBM i command.

Operation value: Type the command to be executed in the Operation Values field. If more space is needed, press **F4** to display a panel for entering the command parameters. Enter the command parameter values.

Example: The following command executes a program call.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				EXECUTE	CALL RPT125

You can enter OPAL variables, such as TEXT, as parameter values.

Robot Reserved Command Variables

Robot Schedule Reserved Command Variables

If you have Robot Schedule installed, you can work with the reserved command variables right from Robot Reports. If you are not familiar with Robot Schedule reserved command variables, see the *Robot Schedule User Guide*.

Robot Schedule reserved command variables contain global system values such as operator on duty, shift hours, and so forth. Robot Schedule reserved command variables are evaluated when the job is run and the resulting values are stored in the variable for use by comparison operations performed by Robot Reports.

Example: The following statement accesses the current value of the Robot Schedule reserved command variable @@RBTVAR and compares it to the value in USRFLD1. If they are equal, the statement is true.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	USRFLD1			EQ	@@RBTVAR

Changing the Value of a Reserved Command Variable

You can also change the value of a Robot Schedule reserved command variable using the operation **RBTCHGRSV**. Define the variable in Robot Schedule first, then you can change it in Robot Reports.

Example: You want to store the grand total of the report SAL456 in a reserved command variable so that it can be compared to totals in other reports. Assuming these are numeric values to compare, the first step is to find the line of the report with the words, “Grand Total.” Then you would store the grand total into a numeric user variable such as USRNBR1. This will remove the edit codes from the information. The last step is to store the contents of USRNBR1 in the specially-named Robot Schedule reserved command variable, “@@SL456TOT,” you had previously defined to Robot Schedule.

Logic Operand	Variable	Line	Column Beg.	End	Operation	Operation Values
IF			25	35	EQ	Grand Total
			75	88	CHGTO	USRNBR1
	USRNBR1				RBTCHGRSV	@@SL456TOT
END						

Run a Robot Schedule Job

If you have Robot Schedule installed, you can use the **RBTBCHUPD** command to change a Robot Schedule job record. Use the EXECUTE operation to execute the RBTBCHUPD command. You can change the job schedule and job setup options and pass command variable values and the current LDA as needed. For more information, see the *Robot Schedule User Guide*.

Operation value: Type the following command in the Operation Values field: ROBOTLIB/RBTBCHUPD and **F4** . The prompt screen for the RBTBCHUPD command is displayed. Enter the command parameter values.

Example: The following operation executes the Robot Schedule RBTBCHUPD command.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				EXECUTE	ROBOTLIB/RBTBCHUPD...
					Press **F4** to see command

To create a new job record, enter the job name–don't just specify a job number. To change an existing job record, enter the job number of the record.

You can enter up to 13 dates in your system date format. Robot Schedule adds these dates to the date object for the job. If no date object exists for the job, Robot Schedule creates one for it and names the new date object the Robot Schedule job name.

Run a Robot Schedule Reactive Job

If you have Robot Schedule installed, the **SNDRBTDTA** operation can send notification of the message or of another event to Robot Schedule. The notification satisfies a prerequisite for a reactive job. For more information on reactive jobs, see the *Robot Schedule User Guide*.

The SNDRBTDTA operation sends a completion status to Robot Schedule. Robot Schedule records that status in the job prerequisite list. If all prerequisites for the job have been met, the reactive job can run.

Operation value: Position the cursor in the Operation Value field and press **F4**. The prompt screen for the SNDRBTDTA command is displayed . Enter the job name and completion status for a user job that has already been entered in the prerequisite list of a Robot Schedule reactive job.

Example: The following statement sends notification of the message to Robot Schedule.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				SNDRBTDTA	Press **F4** to see command

Enter the name of a user job that's already on a Robot Schedule prerequisite list. Also, enter the completion status that was entered for the user job in the prerequisite list.

Sending Messages

PAGE — Send a Pager Message

If you have Robot Alert installed, the PAGE operation executes the Robot Alert command that sends a message to a device or broadcast list. For more information, see the *Robot Alert User Guide*.

Operation value: Position the cursor in the Operation Value field and press **F4**. If you have Robot Alert installed, the prompt screen for the RBASNDMSG command appears. Enter a pager message to be sent by Robot Reports to the specified device or broadcast list.

Example: The following statement sends a device message.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				PAGE	Press **F4** to see the Robot Alert Send a Message panel.

Enter the message you want to send. The message length is limited to the number of characters your vendor will allow. Enter the ID or broadcast list name to which you want to send the message. The ID or broadcast list must be in the device directory.

Press **F10** to see additional parameters.

SNDMSG — Send a Message

The **SNDMSG** operation executes the IBM i command **SNDMSG** to send a message to a user.

Operation value: Position the cursor in the Operation Value field and press **F4**. The Extended Command Entry panel displays. The message to be sent can be entered in the Extended Value field.

Example: The following statement sends a user message.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				SNDMSG	Press F4 to enter message text.

Press **F4** to enter a message in the Extended Values field. Put a single quote (') before and after your message. Enter the name of the user profile after the message. For example:

'The account receipts report will no longer be sent on Mondays' Bob Anderson

If you have Robot Schedule installed, you can press **F7** to see a window listing the Robot Schedule reserved command variables. You can use any of these variables as a parameter value in the note. The current value of the variable is substituted when the note is executed.

Add a Report Note

ADDNOTE — Add a Report Note

The **ADDNOTE** operation adds a public note to a report. This operation should always follow an INCLIN operation.

Important: This function works best when it is used to add only one note per page. Adding a note increases report processing time and storage requirements. Therefore, create notes sparingly.

Operation value: Position the cursor in the Operation Value field and press **F4** to display the Extended Command Entry panel. Enter the note text in the Extended Value field.

Example: The following statement adds a public note to the report.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				ADDNOTE	Press F4 to enter note text.

Call a User Program

Use the CALLRP operation to call user programs from the OPAL code. You can pass report data to the user program and back to the OPAL program.

CALLRP — Pass the Report Data to a Program

The **CALLRP** operation calls a user program and passes it the report data in the parameter REPVAR.

Operation value: The program name or the library name/program name.

Example: The following operation passes the report data to a program MYPRG in the current library list.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				CALLRP	MYPRG

REPVAR Parameter

The user program must define the REPVAR parameter as an external data structure as follows:

```
IREPVAR E DS                1024
C *ENTRY        PLIST
C               PARM        REPVAR
```

The REPVAR format is as follows. In general, the values are those described for OPAL variables.

Field	Type	Size	Dec	In	Out	Text
RV1STL	A	1		1	1	First line of report
RVLSTL	A	1		2	2	Last line of report
RVLLIN#	P	4	0	3	5	Current line number
RVPAG#	P	9	0	6	10	Current page number
RVJOB	A	10		11	21	Job name
RVJOB#	A	6		21	26	Job number
RVUSER	A	10		27	36	Job user
RVSYS	A	8		37	44	System name
RVTXT1	A	132		45	176	Line of text – Part 1
RVTXT2	A	246		177	422	Line of text – Part 2
RVSDAT	P	6	0	423	426	System date
RVSTIM	P	6	0	427	430	System time
RVWORK	A	1		431	431	Is it a work day?
RVDAY#	P	1	0	432	432	Day number
RVDMTH	P	2	0	433	434	Day of the month
RVWEEK	P	1	0	435	435	Week number
RVLDAY	P	6	0	436	439	Last day of the month
RVFORM	A	10		440	449	Form type
RVUDTA	A	10		450	459	User data
RVSTS	A	1		460	460	Include status
RVOUTQ	A	10		461	470	Output queue
RVQLIB	A	10		471	480	Output queue library
RVSPL#	A	6		481	486	Spool file number
RVRTPG	A	1		487	487	Retain paging
RVOVRF	P	3	0	488	489	Overflow line number

Field	Type	Size	Dec	In	Out	Text
RVLSHD	P	3	0	490	491	Heading lines / Page
RVLPAG	P	9	0	492	496	Last page number
RVPSZW	P	3	0	497	498	Page width
RVRSET	A	10		499	508	Report set name
RVPRTF	A	10		509	518	Report name
RVSEGN	A	10		519	528	Report segment name
RVPSZL	P	3	0	529	530	Page length
RVSPLF	A	10		531	540	Spooled file name
RVRPNM	P	9	0	541	545	Relative page number

Go to Tagged Location

Use the TAG operation to assign a name to a location in the program, and then use a GOTO operation to continue processing at that location. The * operation lets you add comments to the program.

The available operations for tagging a location:

- TAG
- GOTO
- *

These operations are described in the following sections.

TAG—Tag a Program Location

The **TAG** operation assigns a name to a location in the program. A GOTO operation can then specify the tag name to go to the tagged location.

Operation value: The name for the location.

Example: The following statement assigns the name BEGIN to this location in the OPAL code.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				TAG	BEGIN

GOTO—Go to Tagged Location

The **GOTO** operation transfers processing to the location specified by the tag name. When a GOTO is performed, processing immediately “jumps” to the tagged location. Thus, the next statement processed is the statement that follows the TAG operation.

Operation value: The name specified on a TAG operation in the code.

Example: The following statement continues processing at the TAG statement that contains the name BEGIN.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				GOTO	BEGIN

*—Add a Comment

The * (asterisk) operation indicates that the statement is a comment used to document the processing performed by the program. You can enter text in the Operation Value field to describe what the code does, or you can leave the field blank to improve the readability of the code.

Operation value: Any characters.

Note: A comment cannot appear between an IF and an END statement.

Example: The following statement inserts a blank line in the code for readability.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				*	

GOTO Example

The following is an example of OPAL code that uses the TAG, GOTO, and * operations. The code repeats until either Bill or Rob replies to the message.

Logic Operand	Variable	Operation	Operation Values
		TAG	BEGIN
		*	
		*	Bill gets 5 minutes to answer.
		REDIRECT	BILL
		RPYWITHIN	300
		*	
		*	Rob gets 5 minutes to answer.
		REDIRECT	ROB
		RPYWITHIN	300
		*	
		*	Go back to Bill.
		GOTO	BEGIN

Operation Values

Most OPAL conditions and operations require a value in the Operation Values field. The value can be represented by a constant or by an OPAL variable.

Example: The first statement below compares the value in columns 15 to 20 with the constant DEPT01. The second statement compares the values of two OPAL variables – PAGENBR and LASTPAGE:

Logic Operand	Variable	Line	Column Beg.	Column End	Operation	Operation Values
IF			15	20	EQ	DEPT01
IF	PAGENBR				EQ	LASTPAGE

How the Column Field is Compared to Numeric Constants and Variables

If you enter a numeric constant or variable in the Operation Value field, Robot Reports assumes that you are looking for numbers in the column field as well. Therefore, letters in the field are ignored with the exception of the letters CR following a number. Robot Reports interprets the CR as meaning this number is a credit and makes the number negative. Robot Reports also recognizes a leading or trailing minus sign (-) as making a number negative. If Robot Reports finds a decimal point in the specified columns, it will keep numbers to the left and right of the decimal point. Otherwise, it treats the value in the columns as a whole number.

Example: The first statement below compares the value in columns 75 to 85 with the constant 900. The second statement compares the value in those columns with the named OPAL variable USRNBR3.

Logic Operand	Variable	Line	Column Beg.	Column End	Operation	Operation Values
IF			75	85	EQ	900
IF			75	85	EQ	USRNBR3

OPAL Constants

You can use a word as a constant even though it's also the name of an OPAL variable. To do so, enclose the word in single quotation marks ('). For example, LINE is an OPAL variable so, to use LINE as a constant value, you must enter 'LINE'.

The following are special OPAL constants:

BLANK	Blank characters to fill the field.
YES	True logical value.
NO	False logical value.

OPAL Variable Values

You can use any of the OPAL variables to represent a value. For example, to represent the current time in the Operation Value field, use SYSTIME.

The following is a list of available variables:

BUNDLE	RCPPRF	USRFLD*n*
DAY	RECIPIENT	USRFLG*n*
DAYMTH	REPORTNAME	USRLONG*n*
DISTYPE	REPORTSET	USRNBR*n*
FIRSTLINE	SEGMENT	WEEKNO
FORMTYPE	SPLNAME	WORKDAY
JOB	SPLNBR	
JOBNUMBER	STSINC	
LASTDAY	SYSDATE	
LASTLINE	SYSTEM	
LASTPAGE	SYSTIME	
LINENBR	TEXT	
OUTQ	USER	
OUTQLIB	USRCNT*n*	
PAGENBR	USERDTA	

User Variables

The OPAL variables USRCNT*n*, USRFLD*n*, USRFLG*n*, USRLONG*n*, and USRNBR*n* are user variables. A user variable does not contain a value until the OPAL code assigns a value to it. If you specify a user variable as the operation value, the value stored in the user variable is used by the condition or operation. Do this if the operation value should vary depending on the OPAL statements executed.

For example, the following OPAL code stores in USRFLD1 the name of the person to whom the message is directed so that the Reply OPAL code can send a message to the person who replied to the message.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				TAG	BEGIN
	USRFLD1			CHGTO	DAVECTR

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
				REDIRECT	USRFLD1
				RPYWITHIN	300
	USRFLD1			CHGTO	ROHITCTR
				REDIRECT	USRFLD1
				RPYWITHIN	300
				GOTO	BEGIN

When the reply is received, the name of the message center that replied to the message is in USRFLD1. The Reply OPAL code could then perform a SNDMSG operation that sends a message to the message center in USRFLD1.

Exception Distribution Operations

Under normal distribution, a user will receive a report every time the report is run. There are some users that may not want to receive the report every time it is run. This is when exception distribution should be used. You can code OPAL to tell Robot Reports what days you want to skip distribution. Use the SKIP operation to define when distribution should not occur.

SKIP — Skip Processing

The **SKIP** operation stops the segment from being printed.

Operation Value: None

Example: You don't want to receive a report except on the last day of the month. Enter the following code:

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
IF	SYSDATE			NE	LASTDAY
				SKIP	
END					

Report View Operations

Most report views are created interactively by report recipients from the View a Report panel. Recipients graphically manipulate the layout to suit their purposes. They can then save the view they have created. The view is stored with the report. As a result, report administrators should very seldom

need to code report view OPAL. However, if the report program is changed, you may code global changes to the views with these operations.

When defining a report view, you may wish to freeze portions of the display. For example, you might want the title of the report and column headings to stay in place as you scroll through the report. Or, you might want a specific column to stay in view as you move left and right in a wide report. Robot Reports lets you do all these things.

TITLE — Retain Title On Screen

The **TITLE** operation is used to retain the specified lines of the first page of the report onscreen. The TITLE operation is valid only for online viewing of reports.

Example: The following OPAL code will keep the first line of the first page of the report onscreen.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
		1		TITLE	

FREEZE — Freeze Lines or Columns

The **FREEZE** operation is used to retain either lines or columns in a view. The FREEZE operation is valid only for online viewing of reports. To freeze a line, enter a line number in the Line field and FREEZE in the Operation field. To freeze a column, enter the beginning and ending numbers of the column in the Column Begin End field and FREEZE in the Operation field.

Example: The following OPAL code will freeze line three and columns 70 to 79 of the report.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
		3		FREEZE	
			70 79	FREEZE	

Additional View Operations

OPAL provides three additional operations for formatting report views. These operations allow you to move, copy, and exclude portions of the report from the view. The MOVE, COPY, and EXCLUDE operations are valid both for online viewing of reports and printed reports.

MOVE — Move a Column

The **MOVE** operation allows you to move a column. This operation requires “from” and “to” information. The “from” information is where the column is now. You must enter those column numbers in the Column Begin End field. The “to” information is the column number where the moved column is to start. You enter this number in the Operation Value field.

Example: The following OPAL code will move the information that is presently in columns 70 to 79 to columns 40 to 49. Information that was formerly in columns 40 to 49 is shifted to the right.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
			70 79	MOVE	40

COPY — Copy a Column

The **COPY** operation allows you to make a copy of a column. You enter which columns to copy in the Column Begin End field and where to place the copy in the Operation Value field.

Example: The following OPAL code will copy the information that is currently in columns one to five and place the copy in columns 128 to 132.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
			1 5	COPY	128

EXCLUDE — Exclude from View

The **EXCLUDE** operation allows you to exclude columns from a view. You may wish to exclude columns that contain unneeded information or information that the report recipient is not authorized to see.

Example: The following OPAL code will suppress the display or printing of the information in columns 51 to 57.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
			51 57	EXCLUDE	

View Operations for Downloading

OPAL provides two special operations for formatting report views that will be downloaded to a workstation and manipulated in spreadsheet programs. These operations, **ALPHACOL** and **NUMBERCOL**, are used to designate which columns should be treated as alphabetic information and which should be treated as numeric information for downloading purposes. Only the columns of a report that are designated as alphabetic or numeric are downloaded with the following exceptions:

- If no columns have been marked, the entire report is downloaded as a single alpha column.
- Any lines marked as titles or frozen are not downloaded.

Creating a view for downloading is explained in the *Robot Reports Recipient Guide*. Use report viewing to designate columns for downloading, and Robot Reports will create the OPAL code for you.

ALPHACOL — Mark an Alphabetic Column

The **ALPHACOL** operation allows you to designate columns of alphabetic information for downloading into spreadsheet programs. Dates should be designated as alpha columns. When alpha columns are

downloaded, the data is enclosed in double quotation marks. These quotation marks will not appear in your spreadsheet. They simply tell the spreadsheet that the field is an alpha string. Leading and trailing blanks are retained.

Operation value: None.

Example: The following OPAL code will mark the information in columns 8 to 25 as alphabetic for downloading.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
			8 25	ALPHACOL	

NUMBERCOL — Mark a Numeric Column

The **NUMBERCOL** operation allows you to designate columns of numeric information for downloading into spreadsheet programs. When columns designated as numeric are downloaded, all the edit symbols are removed except for negative signs and decimal points. Leading and trailing blanks are deleted. The numeric information is not enclosed by any symbols.

Operation value: None.

Example: The following OPAL code will mark the information in columns 27 to 38 as numeric for downloading.

Logic Operand	Variable	Line	Column Beg. End	Operation	Operation Values
			27 38	NUMBERCOL	

Examples

OPAL Code Examples

This section contains examples of OPAL code that you could use with your reports. These are examples only. Review the code carefully; you may need to modify it for use on your system. **Note:** When entering a long command on the IBM i, type the basic command (such as VRYCFG) and press **F4** to open a screen where you can enter the parameters.

OPAL Example 1: Include a Specific Page Only

The following examples show how you can process a spooled file and include only a specific page in your report. They demonstrate the value of using the QUIT and QUITPAGE operations to improve the performance of the OPAL code.

The QUITPAGE operation stops reading the remaining lines of a page (of a given segment) after it determines that the page does not match the criteria you specified. The next page starts processing. If you didn't use QUITPAGE, OPAL would process every line of the spooled file, decreasing performance.

The QUIT operation stops the processing of the spooled file for the current segment. This is useful if you know that the spooled file has a specific section you want to segment. Thus, in the second example, since you want only the 10th page of the report, using QUIT after you segment it improves performance.

The first example distributes the last page of the report only. It checks to see whether the page being processed is the last page in the report. If it is, it includes the page and quits processing the page.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF	PAGENBR				EQ	LASTPAGE	10
					INCPAG		20
END							40
							50
							60
							70
							80

The second example distributes only the 10th page of a report. It checks to see whether the page being processed has a page number of 10. If so, it includes the page and quits processing the spooled file. If the page being processed is not page 10, it quits processing the page and goes to the next page.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF	PAGENBR				EQ	10	10
					INCPAG		20
					QUIT		40
END							50
					QUITPAGE		60
							70
							80

OPAL Example 2: Building a Report Segment

This example builds a report segment for a product manager. The report segment includes only the parts that did not achieve the desired margin of 35% or more. First, compare for “Part No” to make sure you have the right print lines. Then, compare the gross margin percent column to 35 to see if the line should be included. Because only a limited number of lines will be included, select **Retain Paging=No** on the Report Segment panel, and Robot Reports will renumber the pages of this report segment.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			12	18	EQ	Part No	10
AND			88	90	LT	035	20
					INCLIN		30
END							40
							50
							60
							70

OPAL Example 3: Include Page — "Best Bet" OPAL

The most efficient OPAL will make its decision as early as possible when reading a report and perform a QUITPAGE so it does not process needless lines of the report page. This example processes 2 lines on each page of the report before reaching a decision. If the report is 200 pages long, the OPAL will process 400 times. If the decision line were line 10, the OPAL would process 2000 times for the same 200 pages, thus extending the processing time of the report.

The example uses the QUITPAGE operation (rather than QUIT) to decrease processing time. Since Dept 100 can appear on any page, the OPAL must read each page of the spooled file to determine whether it is to be included.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF					EQ	2	10
IF			20	30	EQ	Dept #100	20
					INCPAG		30
					QUITPAGE		40
ELSE							50
					QUITPAGE		60
END							70
END							80
							90

OPAL Example 4: Simple Execute Example

This example shows a simple comparison to determine if a stock item is below a certain level. If the total number of items is less than 100, the OPAL executes the SNDMSG command requesting an order to be placed. You can execute any iSeries command using OPAL.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			15	21	EQ	Total	10
IF			26	29	LT	100	20
					EXECUTE	SNDMSG MSG('Stock level is...	30
END							40
END							50
							60
							70

Note: The full command for Seg 30 is as follows:

SNDMSG MSG('Stock level is below 100. Please reorder within 24 hours') TOUSR(JERRY)

OPAL Example 5: Simple Paging Example

This example is similar to Example 4. It shows a simple comparison to determine if a stock item is below a certain level. If the total number of items is less than 100, the OPAL sends a message using Robot Alert.

```
 Logic                  Column
Operand  Variable Line Beg End  Operation   Operation Values                  Seq
 IF                    15  21   EQ          Total                             10
 IF                    26  29   LT          100                               20
                               PAGE        ** Press F4 to see command**_     30
 END                                                                           40
 END                                                                           50
                                                                               60
```

The following is what you'll see if you press **F4**:

```
                    Send a Message to a Device (RBASNDMSG)

Type choices, press Enter.

Message Text . . . . . .   'Stock level is below 100. Reorder ASAP'
                                                                              ...
Device ID or Broadcast List  . .   JEFF          Character value
Response Required  . . . . . .     *SETUP        *SETUP, *YES, *NO
Truncate Message for Vendor  . .   *YES          *YES, *NO

                                                                        Bottom
F3=Exit   F4=Prompt   F5=Refresh   F10=Additional parameters   F12=Cancel
F13=How to use this display        F24=More keys
```

OPAL Example 6: Using Robot Schedule Calendars

These examples distribute a report by comparing the system date to the date in the Robot Schedule calendar, STANDARD. Robot Schedule must exist on your system to use this OPAL code.

In the following example, the report will not be distributed on any day except Friday. Robot Schedule considers Monday as the first day of the week, so the 5th day is Friday.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF	DAY				NE	5	10
					EXCREPORT		20
END							30
							40
							50
							60

In the following example, if the last day of the month is not equal to the system date, the report will not be distributed.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF	LASTDAY				NE	SYSDATE	10
					EXCREPORT		20
END							30
							40
							50

OPAL Example 7: AUTORUN Example

This example shows two totals being compared. If the values in USRNBR1 and USRNBR2 are equal, the OPAL executes the AUTORUN command to run the specified Robot Schedule job.

```
Logic                    Column
Operand  Variable Line Beg End  Operation   Operation Values                  Seq
IF       LASTLINE      15  26   EQ          Y                                 10
IF                     15  26   EQ          Grand Total                       20
                       70  80   CHGTO       USRNBR1                           30
                       85  95   CHGTO       USRNBR2                           40
IF       USRNBR1                EQ          USRNBR2                           50
                                AUTORUN      ** Press F4 to see command **    60
END                                                                           70
```

The following example shows the Autorun – Modify ROBOT job panel, which appears when you press **F4**:

```
                    Autorun - Modify ROBOT job (AUTORUN)

Type choices, press Enter.

ROBOT Job Number . . . . . .   42576
ROBOT Job Name . . . . . .     __________    Character value
Date to run this job . . . .   ______        Character value
Schedule Override  . . . . .   D             D, E, H, N, O, R, S
Time to run this job . . . .   ____          0000-2359
Job Description  . . . . . .   __________    Character value, *CURRENT
Job description library  . .   __________    Character value, *CURRENT
Output Queue . . . . . .       __________    Character value, *CURRENT
Output Queue Library . . . .   __________    Character value, *CURRENT
Job Queue  . . . . . .         __________    Character value
Job Queue Library  . . . . .   __________    Character value
Number of copies of print file _____         1-255
Library List Name  . . . . .   __________    Character value
User Profile . . . . . .       __________    Character value, *CURRENT
Auto Tune Pool Size  . . . .   _______       Character value
Job Switches . . . . . .       ________      Character value, *CURRENT...
                                                                   More...
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

OPAL Example 8: Finding a Line

This example will find a value on a line even though the data might not always print on the same line number. First, you must find something different about the line. In this example, the OPAL will include all pages for Department 29. Although the actual line that begins the Department 29 data may be different each time the report prints, the line will always contain the values “Dept” in columns 5 through 10 and “029” in columns 12 through 15.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			5	10	EQ	Dept.	10
IF			12	15	EQ	029	20
					INCPAG		30
					QUITPAGE		40
END							50
END							60
							70
							80

OPAL Example 9: Multiple Reports with the Same Spool File Comparison Data

Reports sent from a mainframe to an IBM i have the same spooled file comparison data. The only way to tell the difference is to look at the contents of the spooled file. Robot Reports operates the same way. It treats spooled files with the same spooled file comparison data as one report, and one way to distribute the individual reports is to create report segments. OPAL code for each segment searches for the report title (or titles) that identifies the individual report.

An alternative is to direct the mainframe spooled files to an output queue that is not being monitored by Robot Reports. Change the user data on the spooled file to make it unique, and then transfer it to an output queue that is being monitored.

Another option is to use Report Title Comparison Information when setting up your Report Name. Robot Reports looks at the spooled file comparison data, and if all of it is the same, it uses the Report Title comparison information to select which spooled files to process. For more information on Report Title Comparison Information, see the Report Set section in the *Robot Reports Administrator Guide*.

The following example uses an OPAL statement to compare whether the report title is equal to “SOD Statistic Report” and “Division: Eastern”. If both conditions exist, the page is included (INCPAG); if they do not, it excludes the report (EXCREPORT).

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF		5	30	55	EQ	SOD Statistic Report	10
AND		6	16	38	EQ	Division: Eastern	20
					INCPAG		30
					QUITPAGE		40
ELSE							50
					EXCREPORT		60
END							70
							80

OPAL Example 10: Report Segments that Start in the Middle of a Page

This example illustrates creating a report segment for each department even though the report does not have page breaks when the departments change. Using the STRINCLIN operation will print blank lines on the first page of the segment up to the point where you include lines to be printed. From there, the actual report lines will be printed. When you specify the ENDINCLIN operation on the last page, the remaining lines on the page will be blank.

First, use an OPAL statement to compare if the report line is equal to “Department:” starting in column 20. If it is, start including the lines as the lines you want are being processed. The processing is now in line inclusion mode and the STSINC variable is set to YES. If the department name changes, the department lines have ended, an end line inclusion operation is specified, and the segment processing is ended with a QUIT operation.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			20	30	EQ	Department:	10
IF			32	50	EQ	Production Control	20
					STRINCLIN		30
END							40
IF			32	50	NE	Production Control	50
AND	STSINC				EQ	Y	60
					ENDINCLIN		70
					QUIT		80
END							90
END							100

OPAL Example 11: Including Previous Lines

This example shows how you can create a condensed report that includes just the accounts that need followup. The report has the company name on one line, followed by the address and account balance on the next line. If the account balance is greater than $15,000, the code prints the account balance/address line and uses the INCPRVLIN operation to pick up the previous line with the company name on it.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			70	80	GE	15000	10
					INCLIN		20
					INCPRULIN	1	30
END							40
							50
							60

OPAL Example 12: Using More Advanced Selection Criteria

This example shows how OPAL can read a report and, based on the information in the report, execute a command on the system. The report lists all libraries saved in a backup operation and notes those libraries that were not saved successfully. The OPAL first checks if this was a NONSYS backup; if it wasn't, it quits. If the backup was a NONSYS, the OPAL checks for errors in the backup. The operator is notified only once–when the first error is found.

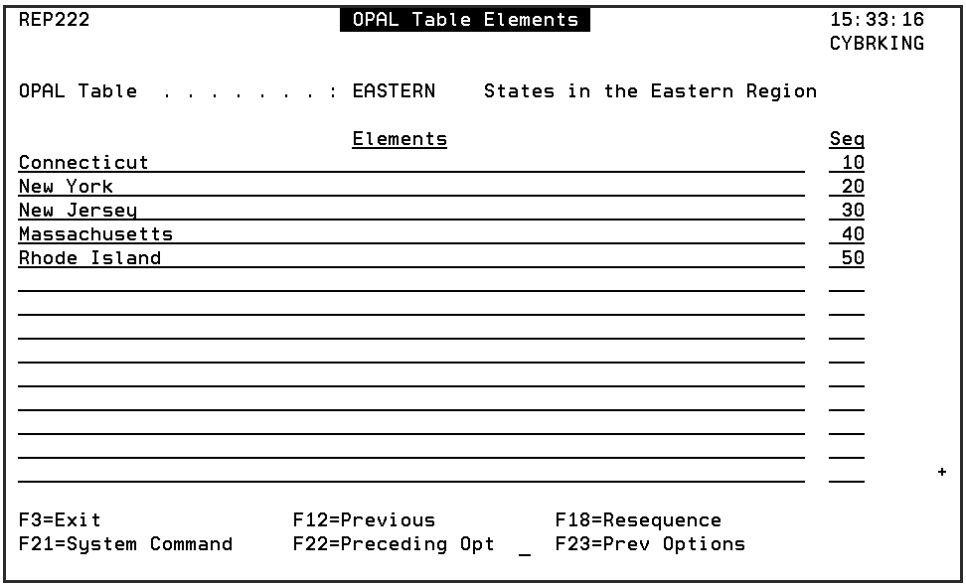
Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF	PAGENBR				EQ	1	10
IF	LINENBR				EQ	6	20
IF			20	30	CT	NONSYS	30
	USRFLG1				CHGTO	Y	40
ELSE							50
					EXCREPORT		60
					QUIT		70
END							80
END							90
END							100
IF			29	45	CT	Errors Occurred	110
AND	USRFLG1				EQ	Y	1 +
	USRFLG1				CHGTO	N	165
					EXECUTE	SNDMSG MSG('Check the backu...	170
END							190

Note: The full command for Seg 170 is as follows:

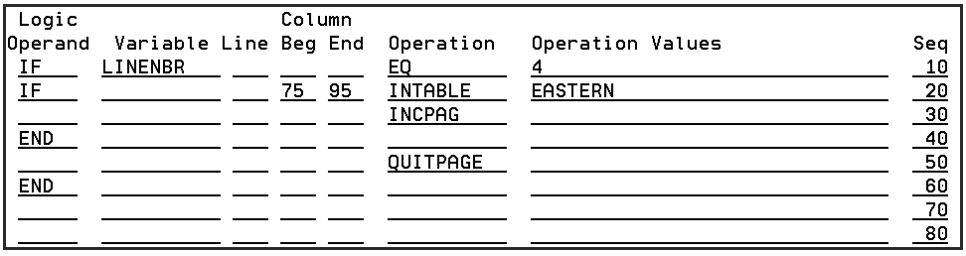
SNDMSG MSG('Check the backup as soon as possible') TOUSR(JEFF)

OPAL Example 13: Using OPAL Tables

This example builds a regional sales report from a national sales report. Each state has a section of the national report. The state name is always printed on line 4 of the report. First, you build an OPAL table that contains the state names that will be included in the report segment called EASTERN. Then, you code OPAL statements to see if the report heading matches a state entry in the table. If it does, include the page.



```
REP222                          OPAL Table Elements                  15:33:16
                                                                     CYBRKING

OPAL Table . . . . . : EASTERN    States in the Eastern Region

                              Elements                                Seq
Connecticut                                                           10
New York                                                              20
New Jersey                                                            30
Massachusetts                                                         40
Rhode Island                                                          50
                                                                      ___
                                                                      ___
                                                                      ___
                                                                      ___
                                                                      ___
                                                                      ___
                                                                      ___
                                                                      ___
                                                                      ___

F3=Exit                 F12=Previous          F18=Resequence
F21=System Command      F22=Preceding Opt _   F23=Prev Options
```



```
 Logic                    Column
Operand  Variable Line Beg End  Operation   Operation Values                Seq
IF       LINENBR       ___ ___  ___ EQ      4                               10
IF       _____         ___ 75   95  INTABLE  EASTERN                         20
_____    _____         ___ ___  ___ INCPAG   _____                           30
END      _____         ___ ___  ___ _____    _____                           40
_____    _____         ___ ___  ___ QUITPAGE  _____                          50
END      _____         ___ ___  ___ _____    _____                           60
_____    _____         ___ ___  ___ _____    _____                           70
_____    _____         ___ ___  ___ _____    _____                           80
```

OPAL Example 14: Creating a Simple Index

Some reports, such as invoices or purchase orders, would be much easier to locate for viewing and reprinting if you could locate them by customer and order number. Indexing allows you to specify certain values that you can use to retrieve a specific invoice for viewing or reprinting. It is important when coding indexing to specify the exact line and page where the index will be found and to use INCPAG to be able to view the entire page when selecting an index entry, and then use QUITPAGE to continue on the next page.

The following example locates the order and customer numbers on line 11 of the report and indexes them both. When line 11 has been processed, it includes the page of the report, quits that page, and continues on the next page.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
					*	Index the Order #	10
IF	LINENBR				EQ	11	20
					INDEX	ORDER#	30
					*	Index the Customer #	40
					INDEX	CUST#	50
					INCPAG		60
					QUITPAGE		70
END							80
							90

OPAL Example 15: Creating a Compound Key Index

The following example creates a compound index by concatenating two different values. It locates the customer and order numbers on line 11 and concatenates them to create the index. After it processes the line, it includes the page of the report and quits the page.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
					*	Index BOTH Customer Number# &	10
					*	the Order#	20
					*	1st..Move the Customer #to a	30
					*	User Field	40
IF	LINENBR				EQ	11	50
			6	13	CHGTO	USRLONG1	60
					*	Next, concatenate the Order#	70
					*	to the same User Field	80
			15	19	BCAT	USRLONG1	90
	USRLONG1				INDEX	Cust#ORD#	100
					INCPAG		110
					QUITPAGE		120
END							1 +

OPAL Example 16: Comparing the Totals of Two Different Reports

This example shows you how to automate the operator duties of comparing a total on one report with a total on another report. If the totals agree, the remaining jobs in the procedure are run.

The following example stores the grand total of the report SAL456 in a Robot Schedule reserved command variable you have created, so it can be compared to the total on report REC459. First, find the line of the report with the words “Grand Total.” Then, store the grand total in a numeric user variable, such as USRNBR1. (This removes any edit codes from the information.) Finally, store the contents of USRNBR1 in the previously-defined Robot Schedule reserved command variable, @@OPAL.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			25	35	EQ	Grand Total	10
			75	88	CHGTO	USRNBR1	20
	USRNBR1				RBTCHGRSV	@@OPAL	30
END							40
							50
							60
							70
							80

Similarly, look for the words “Grand Total” in the report REC459. Change the total to a numeric variable and compare it to the Robot Schedule reserved variable you updated in the SAL456 report. If the totals don’t agree, send a message to Amy.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			38	48	EQ	Grand Total	10
			58	71	CHGTO	USRNBR1	20
IF	USRNBR1				NE	@@OPAL	40
					EXECUTE	SNDMSG MSG('Not in Balance'...	50
					QUIT		51
ELSE							60
					EXECUTE	SNDMSG MSG('Start Mo. End')...	70
					QUIT		71
							80
END							90
END							100

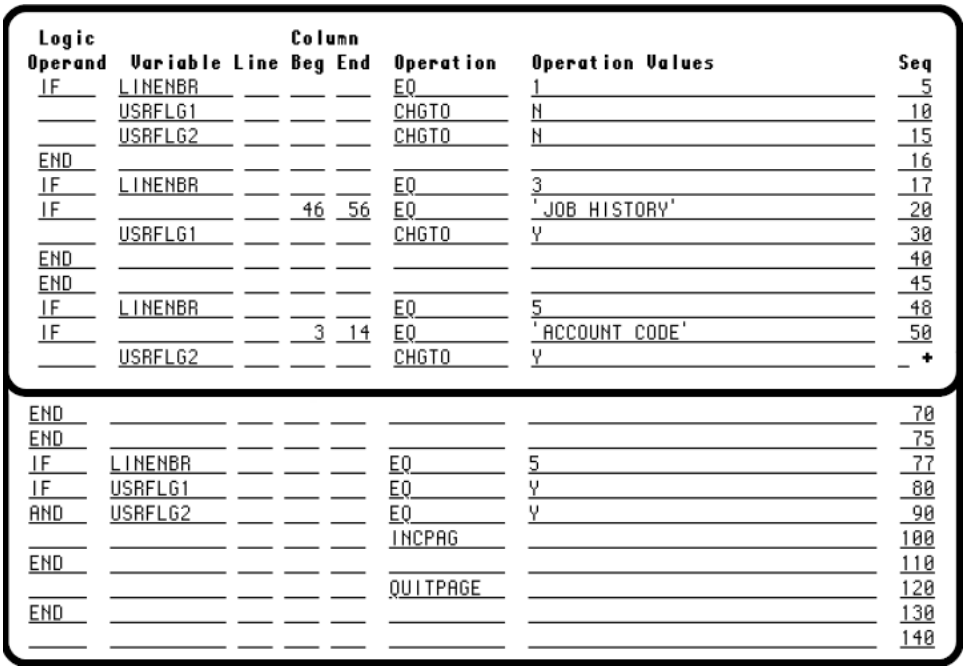
Note: The full commands for Seg 50 and 70 are as follows:

SNDMSG MSG('Not in Balance') TOUSR(AMY)

SNDMSG MSG('Start Mo. End') TOUSR(AMY)

OPAL Example 17: Testing Two Lines and Setting Flags

When Robot Reports processes a report for segmenting, it reads the report one line at a time, so it never reads line 3 and line 5 at the same time. One way to test for a value on two lines is to use flags. This example checks columns 46 through 56 on line 3 for the value JOB HISTORY. If it finds the value, USRFLG1 is set to Y. Then, it checks columns 3 through 14 on line 5 for the value ACCOUNT CODE. If it finds it, USRFLG2 is set to Y. When both flags are set to Y, the page is included.



Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF	LINENBR				EQ	1	5
	USRFLG1				CHGTO	N	10
	USRFLG2				CHGTO	N	15
END							16
IF	LINENBR				EQ	3	17
IF			46	56	EQ	'JOB HISTORY'	20
	USRFLG1				CHGTO	Y	30
END							40
END							45
IF	LINENBR				EQ	5	48
IF			3	14	EQ	'ACCOUNT CODE'	50
	USRFLG2				CHGTO	Y	+
END							70
END							75
IF	LINENBR				EQ	5	77
IF	USRFLG1				EQ	Y	80
AND	USRFLG2				EQ	Y	90
					INCPAG		100
END							110
					QUITPAGE		120
END							130
							140

OPAL Example 18: Testing Two Lines — Shorthand Method

This example is similar to Example 17 in that it reads the values on two different lines and includes or excludes the page based on the results. If the values found on line 1 AND line 3 match the specified criteria, the page is included. If the values do not match, the page is excluded. In both cases, as soon as a decision is made for each page, the OPAL quits processing that page and starts processing the next page.

Logic			Column				
Operand	Variable	Line	Beg	End	Operation	Operation Values	Seq
IF		1	46	56	EQ	JOB HISTORY	10
AND		3	3	14	EQ	ACCOUNT CODE	20
					INCPAG		30
					QUITPAGE		40
ELSE							50
					EXCPAG		60
					QUITPAGE		70
END							80
							90
							100

OPAL Example 19: Highlighting Action Items for Viewing

This example shows how you can use color to highlight action items. Suppose you want to highlight significant past-due accounts. Your aged balance report has 60-day-old amounts in columns 70 to 80 and 90-day-old amounts in columns 85 to 95. You specify that amounts over $15,000 and 60 days old be displayed in yellow, and amounts over $10,000 and 90 days old be displayed in red.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			70	80	GE	15000	10
					HILITE	Yellow	20
END							30
IF			85	95	GE	10000	40
					HILITE	Red	50
END							60
							70
							80

OPAL Example 20: Highlighting Report Segments

This example shows you how to process a report segment and highlight action lines. This differs slightly from using highlighting in a specific report view. When you highlight a report segment, each view is also highlighted. In addition, a report with segment highlighting processes faster than one with view highlighting.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			70	80	GE	500	10
					INCLIN		20
IF			70	80	GE	1000	30
					HILITE	Red	40
END							50
END							60
							70
							80

OPAL Example 21: Checking ASP Status

This example shows how you can use a Robot Schedule job to check the status of an ASP. Create a Robot Schedule job that runs the WRKDSKSTS OUTPUT(*PRINT) command on your system and creates a spooled file. Link Robot Reports to the Robot Schedule job so that it will process the spooled file. The OPAL code checks the spooled file for the percentage of ASP 2 currently being used. If the value is greater than 80 percent, it executes the SNDMSG command to notify someone of the situation.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
					*	Not part of the heading	10
IF	LINENBR				GT	9	20
					*	*** ASP #2 > 80% ***	30
IF			80	80	EQ	2	40
IF			122	123	GT	80	50
					EXECUTE	SNDMSG MSG('ASP2 is over 80...	60
END							70
END							80
							90

Note: The full command for Seg 60 is as follows:

SNDMSG MSG('ASP2 is over 80% full!') TOMSGQ(ARMAND)

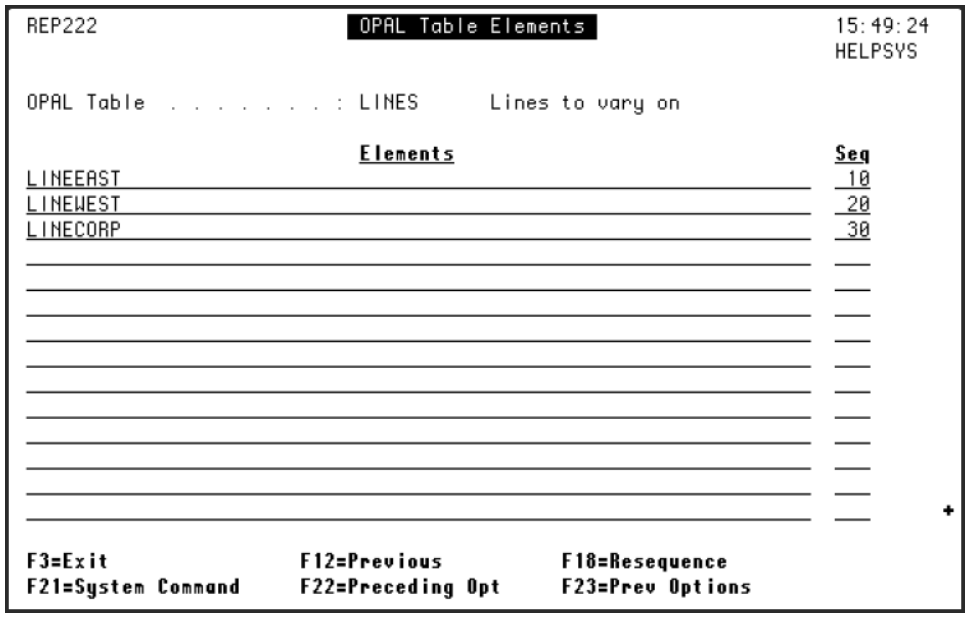
OPAL Example 22: Starting Inactive Writers

This example shows how you can use a Robot Schedule job to start inactive writers. Create a Robot Schedule job that runs the WRKWTR OUTPUT(*PRINT) command on your system and creates a spooled file. Link Robot Reports to the Robot Schedule job so that it will process the spooled file. The OPAL code checks columns 13 to 15 of the spooled file for any writer that has a status of END (not active). When it finds a match, it executes the STRPRTWTR command to start the writer. The OPAL then checks for the last line of the report segment. When it finds the last line, it deletes the segment so it won't be printed or viewed. The original spooled file is deleted only if the field on the report name is set properly.

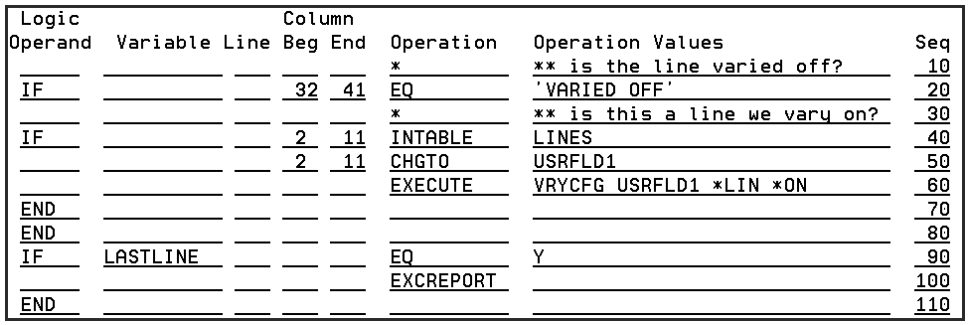
Logic			Column				
Operand	Variable	Line	Beg	End	Operation	Operation Values	Seq
					*	If writer is ended, then start	10
IF			13	15	EQ	'END'	20
			2	11	CHGTO	USRFLD1	30
					EXECUTE	strprtwtr USRFLD1	40
END							50
IF	LASTLINE				EQ	Y	60
					EXCREPORT		70
END							80
							90

OPAL Example 23: Varying On Specific Lines

This example shows how you can use a Robot Schedule job to check the configuration status of a line. Create a Robot Schedule job that runs the WRKCFGSTS OUTPUT(*PRINT) command on your system and creates a spooled file. Link Robot Reports to the Robot Schedule job so that it will process the spooled file. The OPAL code checks all the lines in the spooled file and compares their status to the value “VARIED OFF.” If the line description name that is varied off is in the table LINES, it executes the VRYCFG command to vary on the line. Then, the OPAL checks for the last line of the report segment. When it finds the last line, it deletes the segment so it won’t be printed or viewed. The original spooled file is deleted only if the field on the report name is set properly.



```
REP222                          OPAL Table Elements                15:49:24
                                                                   HELPSYS

OPAL Table . . . . . : LINES      Lines to vary on

                          Elements                                 Seq
LINEEAST                                                           10
LINEWEST                                                           20
LINECORP                                                           30
                                                                   ___
                                                                   ___
                                                                   ___
                                                                   ___
                                                                   ___
                                                                   ___
                                                                   ___
                                                                   ___
                                                                   ___
                                                                   ___
                                                                   ___

F3=Exit                  F12=Previous          F18=Resequence
F21=System Command       F22=Preceding Opt     F23=Prev Options
```



```
Logic                        Column
Operand  Variable  Line  Beg  End  Operation   Operation Values                  Seq
                               *    ** is the line varied off?                 10
IF                             32   41  EQ        'VARIED OFF'                 20
                               *    ** is this a line we vary on?             30
IF                             2    11  INTABLE   LINES                        40
                               2    11  CHGTO     USRFLD1                      50
                                    *   EXECUTE   VRYCFG USRFLD1 *LIN *ON      60
END                                                                            70
END                                                                            80
IF       LASTLINE                  *   EQ        Y                            90
                                    *   EXCREPORT                              100
END                                                                            110
```

OPAL Example 24: Listing Libraries Not Saved

This example shows how you can create a list of libraries that weren't saved during a backup operation. The OPAL code checks the spooled file for the message CPF3774 (Library not successfully saved) and includes the page when it finds a match. Then, it quits processing that page and begins processing the next page.

Use this code on the QPJOBLOG print file created by a Robot Schedule job running a save. Add the command DSPJOBLOG OUTPUT(*PRINT) as the last command in your Robot Schedule job.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF	TEXT				CT	'CPF3774'	10
					INPAG		20
					QUITPAGE		30
END							40
							50
							60
							70

OPAL Example 25: Checking Library Size

This example shows how you can use a Robot Schedule job to determine the size of a library. Create a Robot Schedule job that runs the DSPLIB OUTPUT(*PRINT) command on your system and creates a spooled file. Link Robot Reports to the Robot Schedule job so that it will process the spooled file. The OPAL code checks the library size on the line where it finds the text “Total Size.” If the value is greater than 999999, it executes the SNDMSG command to notify someone of the situation.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			22	31	EQ	'Total size'	10
IF			45	51	GT	999999	20
					EXECUTE	SNDMSG MSG('Library size fo...	30
END							40
							50

Note: The full command for Seg 30 is as follows:

SNDMSG MSG('Library size for TEST1 exceeds 1meg') TOUSR(JANET)

OPAL Example 26: Checking Number of Spooled Files on an Output Queue

This example shows how you can use a Robot Schedule job to check the number of spooled files on an output queue. Create a Robot Schedule job that runs the WRKOUTQ OUTPUT(*PRINT) command on your system and creates a spooled file. Link Robot Reports to the Robot Schedule job so that it will process the spooled file. The OPAL code checks the list of output queues in the spooled file. When it finds an output queue that contains more than 25 spooled files, it executes the SNDMSG command to notify someone of the situation and includes the name of the output queue in the message.

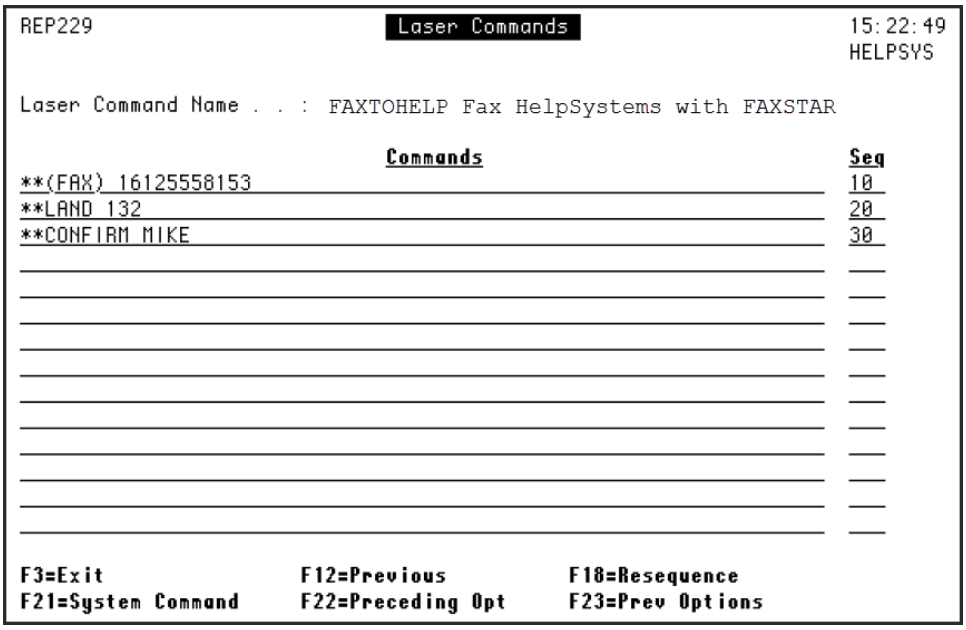
Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
IF			53	55	GT	25	10
			2	11	CHGTO	USRFLD1	20
					EXECUTE	SNDMSG MSG('out1 USRFLD1 ha...	30
END							40
IF	LASTLINE				EQ	Y	50
					EXCREPORT		60
END							70
							80
							90

Note: The full command for Seg 30 is as follows:

SNDMSG MSG('outq USRFLD1 has too many splfs') TOMSGQ(JANET)

Laser Command Example: Sending a Report as a Fax Using FAX*STAR

You do not need OPAL code to send a report as a fax, but you do need to use a Robot Reports laser command. Normally, you put the laser command name in the Report Segment panel and the fax output queue name in the Report Distribution Output Options panel. This laser command example shows the FAX*STAR commands used to dial the fax number, flip the report page to landscape orientation (so the entire page will be transmitted), and send a message to user profile MIKE confirming the fax. Robot Reports will send this report to the output queue with the laser command as the first page ahead of the report.



Exception Distribution Example 1: Distribute on the 15th and 30th of the Month

This example shows how you can use Robot Reports exception distribution to distribute your reports only on certain days of the month. In this example, if the day of the month is not equal to 15 or 30, we skip distribution.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
					*	If the day is not the 15th and	1
					*	not the 30th, then skip	2
IF	DAYMTH				NE	15	10
AND	DAYMTH				NE	30	20
					SKIP		30
END							40
							50

Exception Distribution Example 2: Do Not Distribute on Friday

This example shows how you can use Robot Reports exception distribution to specify that reports not be distributed on a certain day of the week. In this example, if the day is equal to 5 (Friday) we skip distribution.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
					*	Don't distribute the report	1
					*	on Fridays	2
IF	DAY				EQ	5	10
					SKIP		20
END							30

Exception Distribution Example 3: Distribute on the First Monday of the Month

This example shows how you can use Robot Reports exception distribution to distribute your reports on a first weekday of the month. In this example, if the day is not equal to 1 (Monday), or the day is greater than 7, skip distribution.

Logic Operand	Variable	Line	Column Beg	Column End	Operation	Operation Values	Seq
					*	If not Monday - OR	1
IF	DAY				NE	1	20
					*	Date is greater than 7, then	30
					*	skip distribution. Otherwise	40
					*	it is the 1st monday of month	50
OR	DAYMTH				GT	7	60
					SKIP		70
END							80
							90

Quick Reference Guide

Robot Reports Quick Reference Guide

Logic Operands

IF	variable	comparison	value
AND	variable	comparison	value
OR	variable	comparison	value
THEN	variable*	operation	value*
ELSE	–	–	–
	variable*	operation	value*
END	–	–	–

* optional

Comparisons

EQ	Equal	NE	Not equal
CT	Contains	DC	Doesn't contain
GT	Greater than	LT	Less than
GE	Greater than or equal to	LE	Less than or equal to
INTABLE	In OPAL table	NOTINTABLE	Not in OPAL table
INLIST	Recipient in distribution list	NOTINLIST	Recipient not in distribution list

Operations

Operation	Operation Value
*	Comment text
ADD	Amount to increment
ADDNOTE	Note to add to segment
BCAT	Value to concatenate
CALLRP	Robot Reports program to call
CAT	Value to concatenate
CHGTO	New value
DELAY	Number of seconds
ENDEXCLIN	None (ends excluding lines)
ENDEXCPAG	None (ends excluding pages)
ENDINCLIN	None (ends including lines)

Operation	Operation Value
ENDINCPAG	None (ends including pages)
EXCLIN	None (excludes this line)
EXCNXTLIN	Number of lines to exclude
EXCPAG	None (excludes this page)
EXCPRVLIN	Number of lines to exclude
EXCREPORT	None (excludes this report)
EXECUTE	Command to execute
GOTO	TAG statement
HILITE	F4 (select color to highlight)
INCLIN	None (includes this line)
INCNXTLIN	Number of next lines to include
INCPAG	None (includes this page)
INCPRVLIN	Number of previous lines to include
INDEX	Index name for this set of columns
PAGE	F4 (enter message)
QUIT	None (ends all processing)
QUITPAGE	None (quits processing this page)
RBTBCHUPD	F4 (displays RBTBCHUPD panel)
RBTCHGRSV	New value
SKIP	None (skip report distribution)
SNDMSG	F4 (enter message to send to queue)
SNDRBTDTA	F4 (enter user job name and status)
STREXCLIN	None (starts excluding this line)
STREXCPAG	None (starts excluding this page)
STRINCLIN	None (starts including this line)
STRINCPAG	None (starts including this page)
SUB	Value to subtract
TAG	Tag name

Variables

OPAL Variable	Operation Value
BUNDLE	Bundle Code
DAY	Day of week (Mon=1)
DAYMTH	Day number in month
DISTTYPE	Distribution Type
FIRSTLINE	Is this the first line?

OPAL Variable	Operation Value
FORMTYPE	Form type from the report
JOB	Job name
JOBNUMBER	Job number
LASTDAY	Last date of the month
LASTLINE	Is this the last line?
LASTPAGE	Last page number of the report
LINENBR	Line number
OUTQ	Output queue where report is
OUTQLIB	Output queue library
PAGENBR	Page number
RCPPRF	Recipient user profile
RECIPIENT	Recipient name
REPORTNAME	Report name
REPORTSET	Report set
SEGMENT	Segment name
SPLNAME	Spooled file name
SPLNBR	Spooled file number
STSINC	Currently in Include Line mode?
SYSDATE	System date
SYSTEM	System name
SYSTIME	System time
TEXT	Text from the report line
USRCNT*n*	User-defined numeric (*n* =1-5)
USRDATA	User data from the report
USRFLD*n*	User-defined field (*n* =1-5)
USRFLG*n*	User-defined flag (*n* =1-5)
USRLONG*n*	Longer character variable (*n* =1-5)
USRNBR*n*	User-defined numeric (*n* =1-5)
WEEKNO	Week number
WORKDAY	Workday or not