JCL

Chapter a6
Conditional processing

Chapter a1. Introduction to JCL

Chapter a2. Coding JOB statements

Chapter a3. Coding EXEC statements

Chapter a4. Coding DD statements

Chapter a5. Analyzing job output

Chapter a6. Conditional processing

Chapter b1. Using special DD statements

Chapter b2. Introducing procedures

Chapter b3. Modifying EXEC parameters

Chapter b4. Modifying DD parameters

Chapter b5. Determining the effective JCL

Chapter b6. Symbolic parameters

Chapter c1. Nested procedures

Chapter c2. Cataloging procedures

Chapter c3. Using utility programs

Chapter c4. Sample utility application

Chapter a6

Conditional processing

Conditional processing.

Unit introduction.

Be able to:

- **Steps in a job can be conditionally executed by using the IF/THEN/ELSE/ENDIF statement construct.**

- **The IF/THEN/ELSE/ENDIF statement construct allows the execution of job steps based on return codes, abend conditions, system completion codes from a previous job or procedure step.**

- **This construct provides greater functionality than the COND statement and is easier to use, read and understand.**

Course objectives.

Be able to:

- **Identify the various types of job conditions that an IF/THEN/ELSE/ENDIF statement construct can test.**
- **Code IF/THEN/ELSE/ENDIF statement constructs.**
- **Correct invalid IF/THEN, ELSE, and ENDIF JCL statements.**
- **State reasons why IF/THEN/ELSE/ENDIF JCL errors occur.**
- **State the advantages of using the IF/THEN/ELSE/ENDIF statement construct instead of the COND parameter.**

IF/THEN/ELSE/ENDIF construct.

Syntax for the IF/THEN/ELSE/ENDIF statement construct.

What is the syntax for an IF/THEN/ELSE/ENDIF statement construct?

The syntax for coding an IF/THEN/ELSE/ENDIF statement construct is:

```
//name    IF     (relational-expression)   THEN
//name    JCL    statements to be executed when relational-expression is true
//name    ELSE
//name    JCL    statements to be executed when relational-expression is false
//name    ENDIF
```

The IF/THEN/ELSE/ENDIF statement construct can be coded anywhere in the job after the JOB statement.

The Name field coding rules.

Even though the name field is optional, if one is coded then it must follow the normal coding rules for names in JCL such as:

- The name must begin in position 3. If you do not code a name then leave the position blank.
- The name must be unique within the job.
- The first character of the name has to be alphabetical or national and it cannot be a number.
- The remaining characters can be alphanumeric or national.
- The name field must be followed by at least one space.

Valid JCL Names

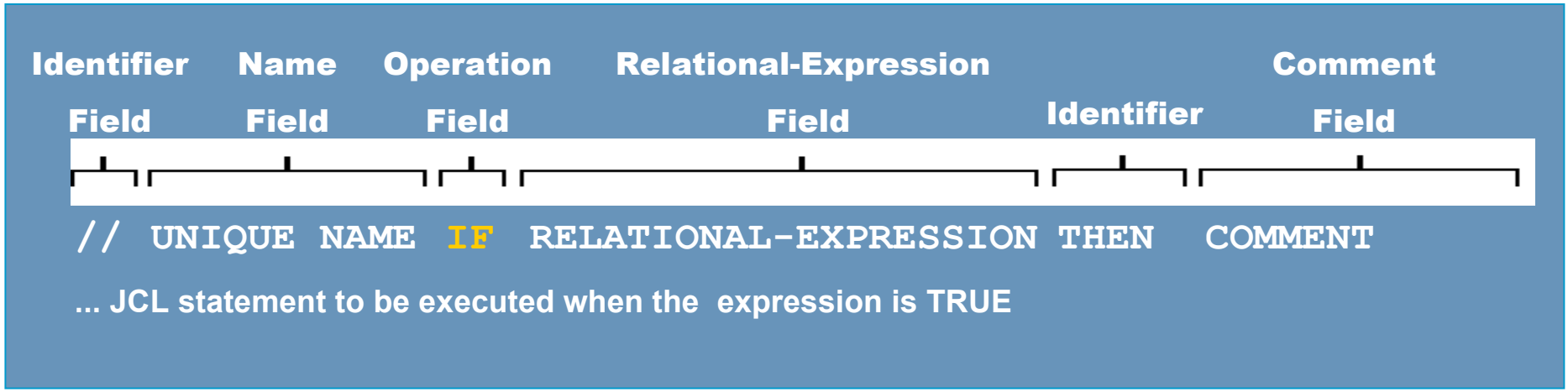
```
//CHECK1 IF (relational-expression) THEN
//UNIQUE IF (relational-expression) THEN
//PROG#1 IF (relational-expression) THEN
```

Invalid JCL Names

`// CHECK1`	(Does not begin in position 3)
`//TOOMANYCHARS`	(More than eight characters)
`//CHECK#2IF`	(Needs blank space after name)

IF/THEN/ELSE/ENDIF construct.

The Operation field – the IF statement.



What is the Operation Field?

The operation field contains the operators IF, ELSE, or ENDIF.

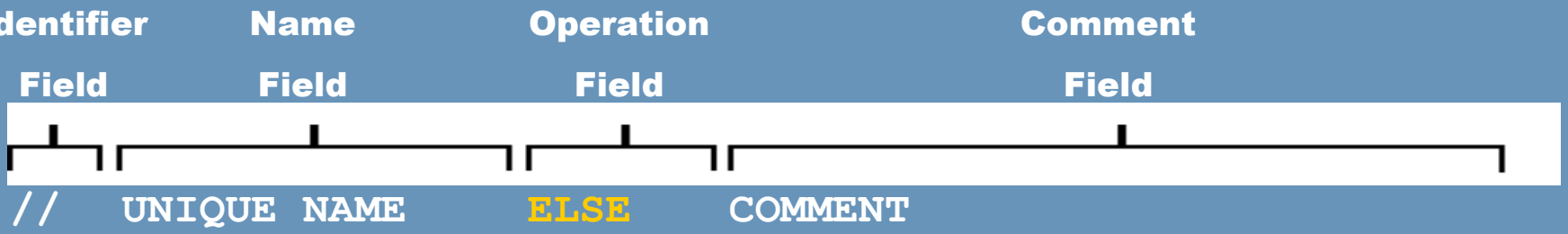
What are the characteristics of the IF statement?

The IF statement always precedes a relational-expression and the identifier THEN.

Following the IF statement are all of the JCL statements to be executed when the relational-expression is true. If there are none, then the IF statement should be followed immediately by the ELSE statement.

IF/THEN/ELSE/ENDIF construct.

The Operation field – The ELSE statement.



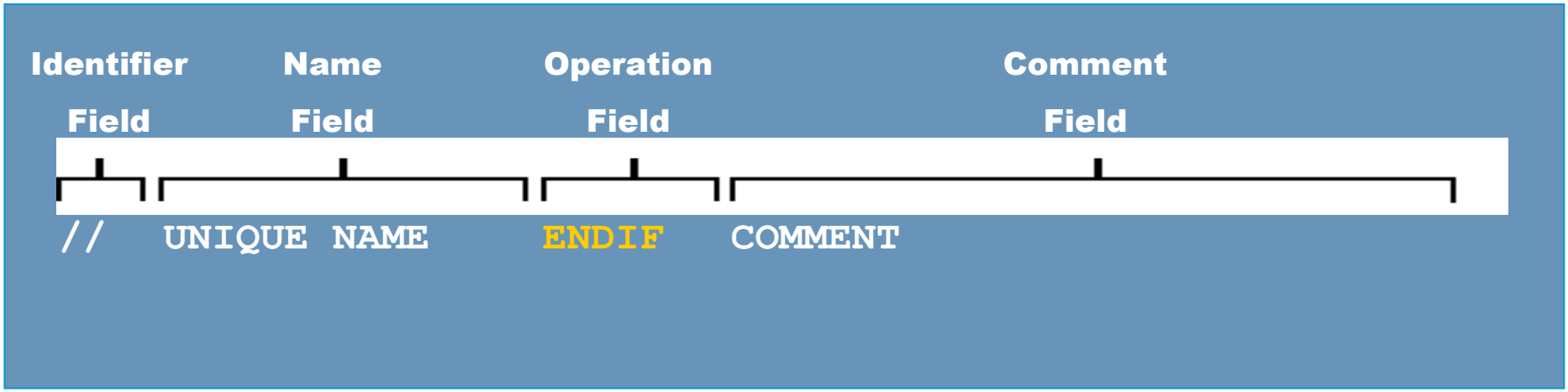
What are the characteristics of the ELSE statement?

Following the ELSE statement are all the JCL statements to be executed when the relational-expression is false. If there are none, then the ELSE statement can be omitted.

The ELSE statement has no parameters.

Anything following the ELSE operator is considered a comment.

The Operation field – the ENDIF statement.



What are the characteristics of the ENDIF statement?

The required ENDIF statement signifies the end of the IF/THEN/ELSE/ENDIF statement construct.

There must be at least one EXEC statement following either the IF statement or the ELSE statement.

Anything coded after the ENDIF statement is considered a comment by the operating system.

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Place the conditional statements in the proper order.

A. // ENDIF

B. //COND IF ABC>5 THEN

C. //STEP2 EXEC PGM=DELFILE

D. // ELSE

E. //STEP1 EXEC PROC=PRINT

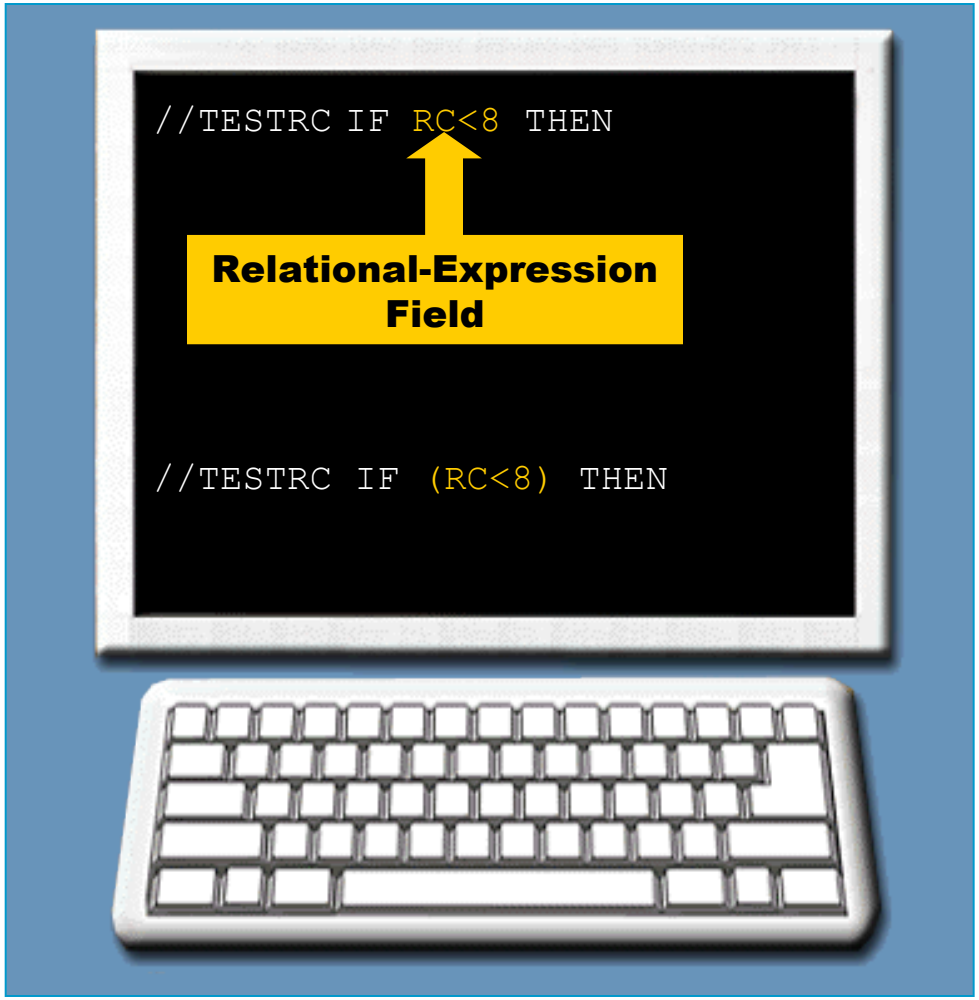
The Relational-Expression field.

What is the relational -expression field?

The relational-expression field follows the IF statement and specifies the condition that is evaluated at execution.

Depending on the values in the expression, the result of the condition is either true or false.

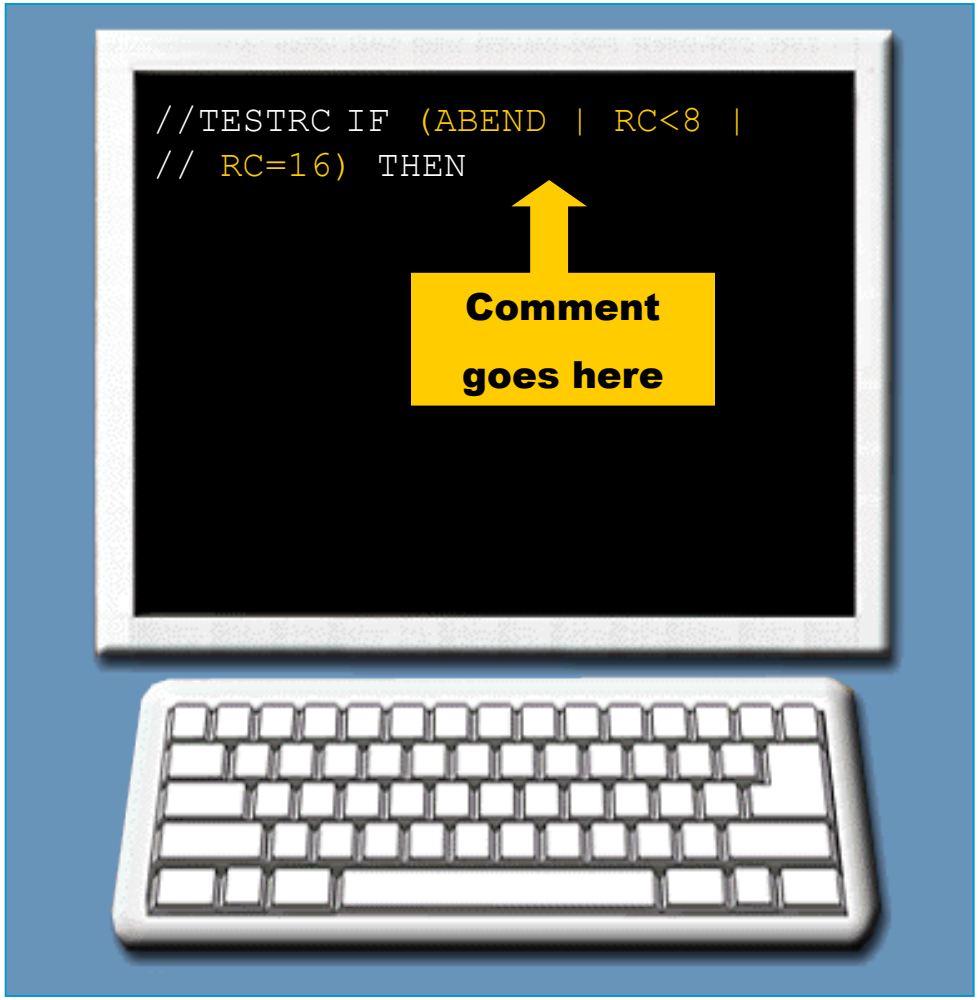
In the example, the first statement tests for a RC of less than 8. Hence the relational-expression is RC<8.



The Relational-Expression field.

Relational-expressions can be continued on more than one line.

To continue the expression, break the expression on a valid blank space and continue on the next line using columns 4 through 16.



The Relational-Expression field.

A relational-expression can consist of any of the following, alone or in combination:

- **Comparison operators.**
- **Logical operators.**
- **NOT operators.**
- **Relational-expression keywords.**

Comparison operators.

What are the characteristics of a comparison operator?

Comparison operators compare a relational-expression keyword to a numeric value. The result of the comparison is either true or false.

The comparison operators are either alphabetic or arithmetic.

Operator	Meaning
GT or >	Greater than
GE or >=	Greater than or equal to
NG or ¬>	Not greater than
EQ or =	Equal to
NE or ¬=	Not equal to
LT or <	Less than
LE or <=	Less than or equal to
NL or ¬<	Not less than

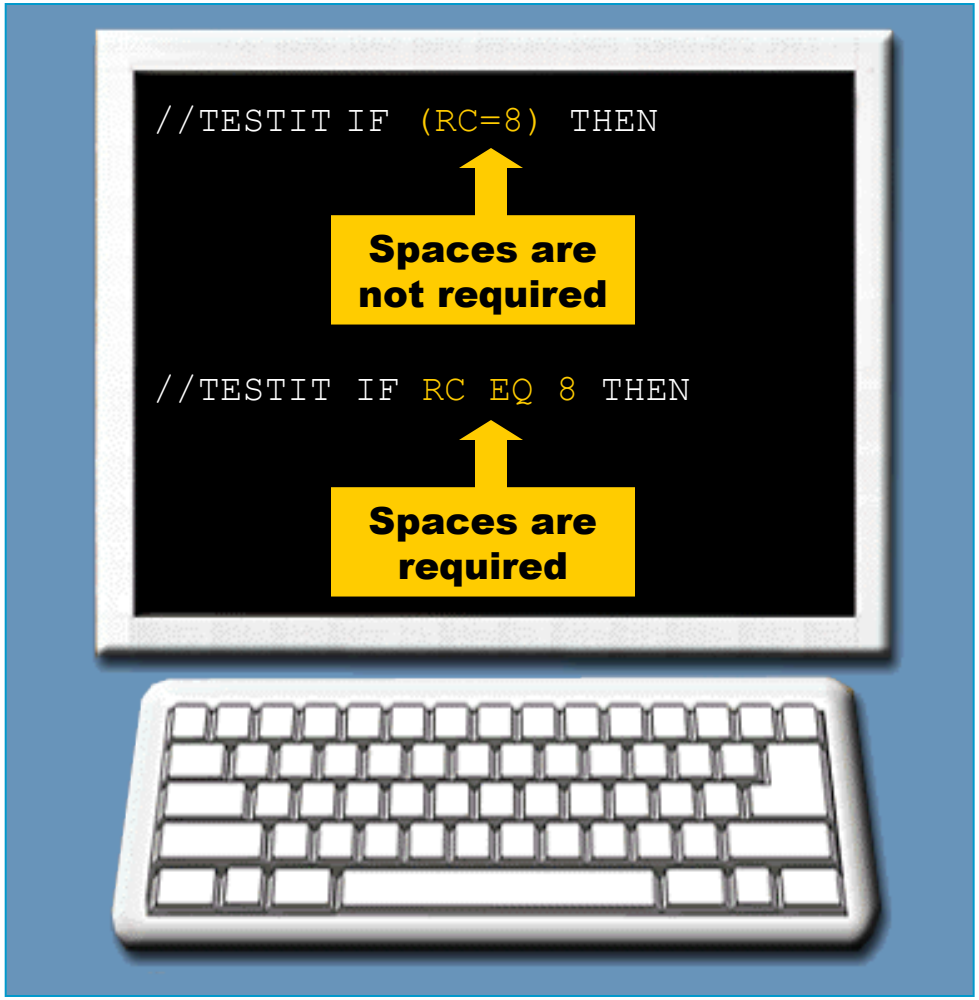
Comparison operators – an example.

In this example, the statement tests if a RC is equal to 8.

The relational-expression (RC=8) must be both preceded and followed by at least one space.

No spaces are required before or after an arithmetic operator, such as = or >.

At least one space is required both before and after alphabetic comparison operators, such as EQ or GT.



IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the JCL statement up to THEN operator to check if the return code is equal to 8, using parentheses.

//TESTIT _____

IF/THEN/ELSE/ENDIF construct.

Logical operators.

The logical operators include:

AND (&)

OR (|)

NOT (¬)

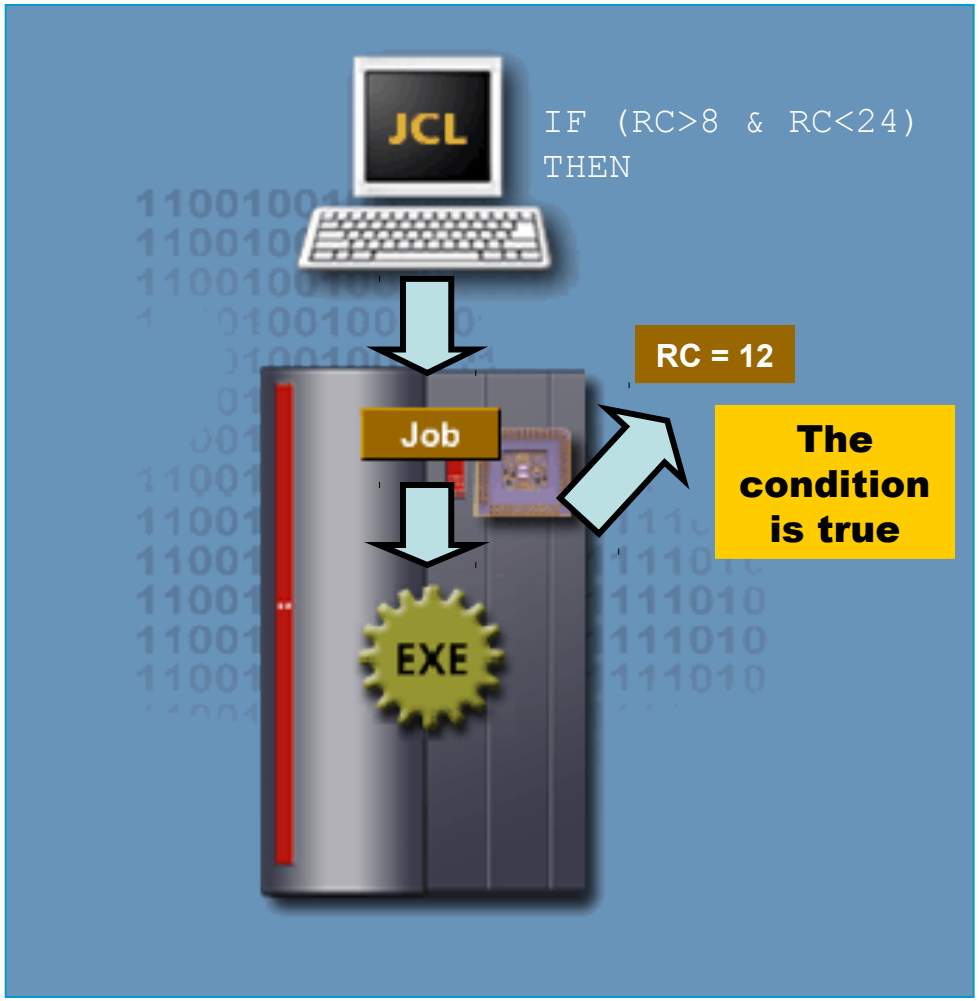
Logical operators - AND.

What are the characteristics of the AND operator?

The AND (&) operator returns a true value only if both relational -expressions are true.

For example, to test if a return code is between 8 and 24:

```
//TEST1 IF (RC>8 & RC<24) THEN
```



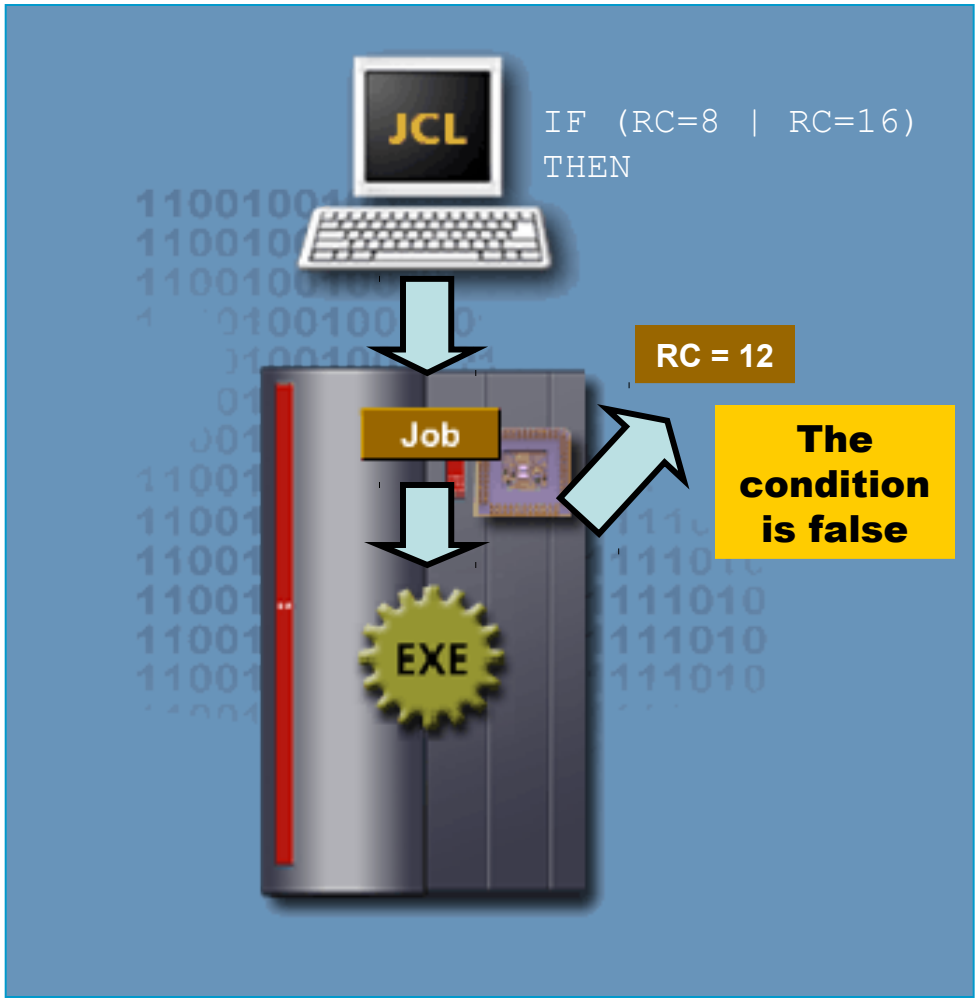
Logical operators - OR.

What are the characteristics of the OR operator?

The OR (|) operator returns a true value if either of the relational-expression is true.

For example, to test if a return code is either equal to 8 or 16:

```
//TEST2 IF (RC=8 | RC=16) THEN
```



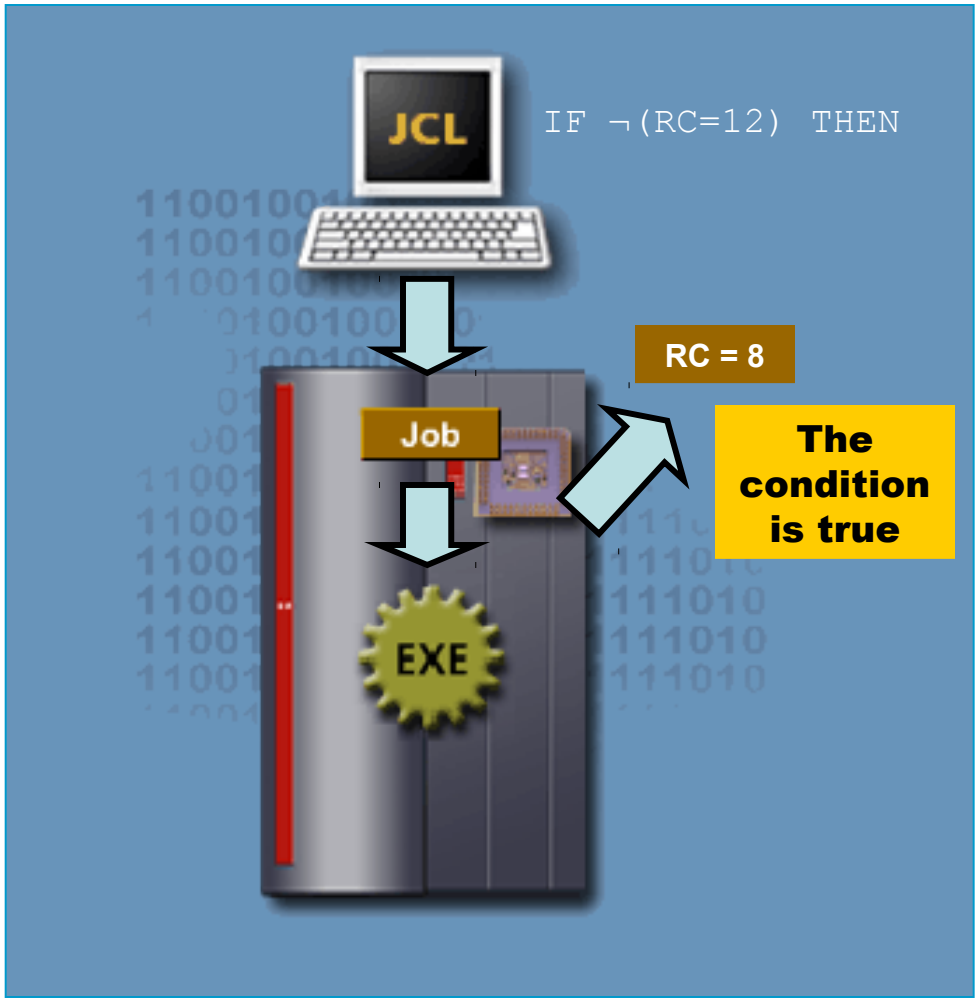
Logical operators - NOT.

What are the characteristics of the NOT operator?

The NOT (¬) operator reverses the testing of a relational-expression. The system evaluates the NOT operator before any comparisons or logical operators.

For example, to test if a return code is not equal to 12:

```
//TEST3 IF ¬(RC=12) THEN
```



IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the AND (&) operator to test if the return code is between 8 and 24.

```
//TEST1    IF (_____________________) THEN
```

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the OR (|) operator to test if the return code is equal to 8 or 16.

```
//TEST2    IF (_____________________) THEN
```

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the NOT (¬) operator (with parentheses) to test if the return code is not greater than 8 and not less than 24.

//TEST3 IF _____ THEN

Relational-Expression keywords.

What are the characteristics of relational-expression keywords?

Relational-expression keywords are used to test a RC, abend condition or abend completion code, or to test if a step began executing. The relational-expression keywords are:

- **RC**
- **ABEND**
- **¬ABEND**
- **ABENDCC**
- **RUN**
- **¬RUN**

Preceding the keyword with a step name relates the expression to a specific job step.
Syntax: **stepname.keyword**

Preceding the keyword with both a step name and procedure step name relates the expression to a specific procedure step.
Syntax: **stepname.procstepname.keyword**

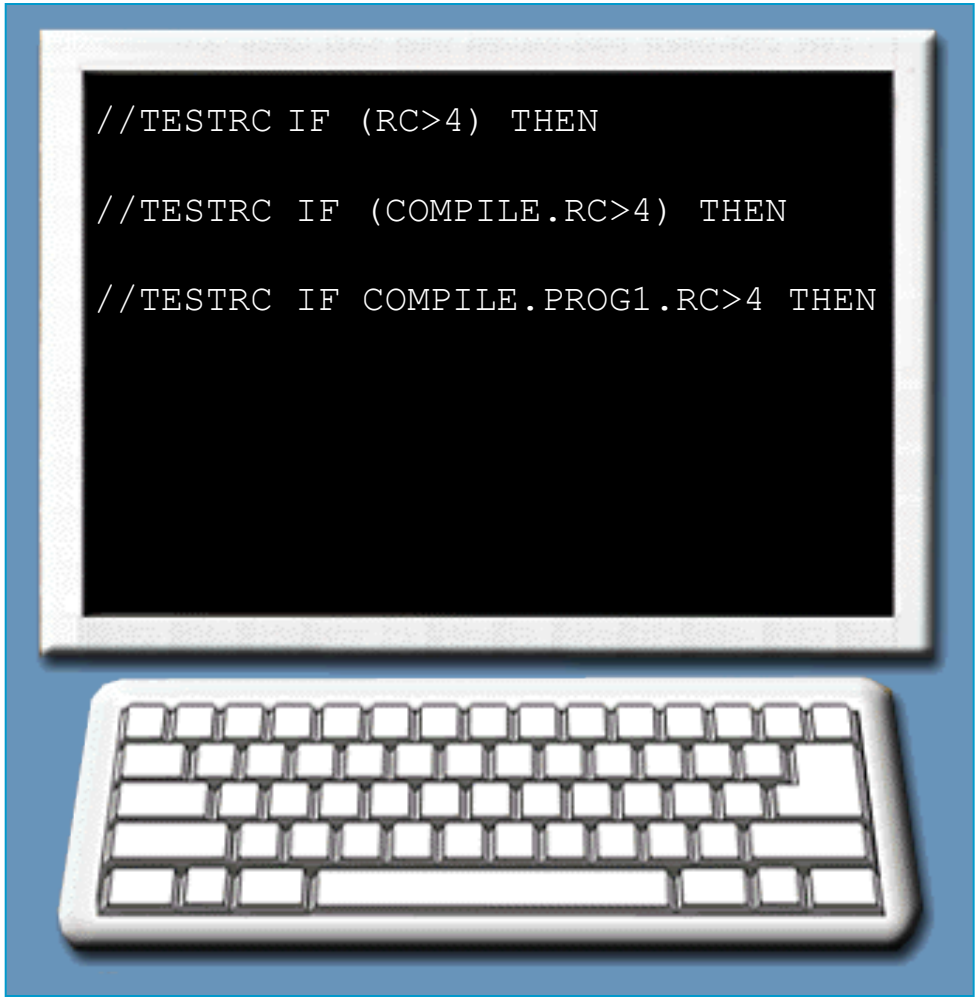
Relational-Expression keywords - RC.

RC represents the highest return code received from a previous job step.

In the example, the first statement checks if the previous job step had a return code > 4.

The second statement tests if a prior job step named COMPILE produced a return code > 4.

The third one checks if a specific procedure step, PROG1 in the job step COMPILE, produced a return code > 4.



Relational-Expression keywords – ABEND.

What are the characteristics of the ABEND keyword?

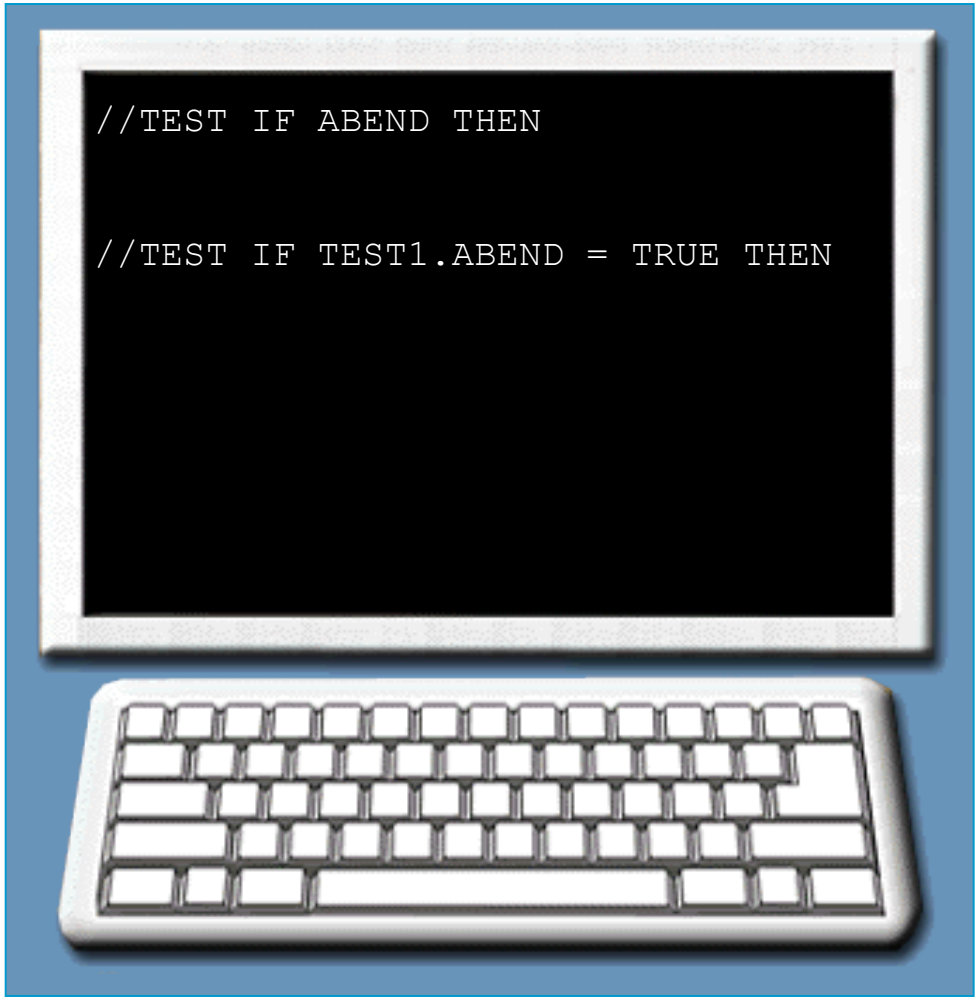
The keyword ABEND tests for abnormal termination from any previous job step. The syntax used for ABEND is:

```
//name IF ABEND THEN
```

or

```
//name IF ABEND = TRUE THEN
```

Both these statements will test for an abnormal termination in any of the previous steps.



Relational-Expression keywords - ¬ABEND.

What are the characteristics of the ¬ABEND keyword?

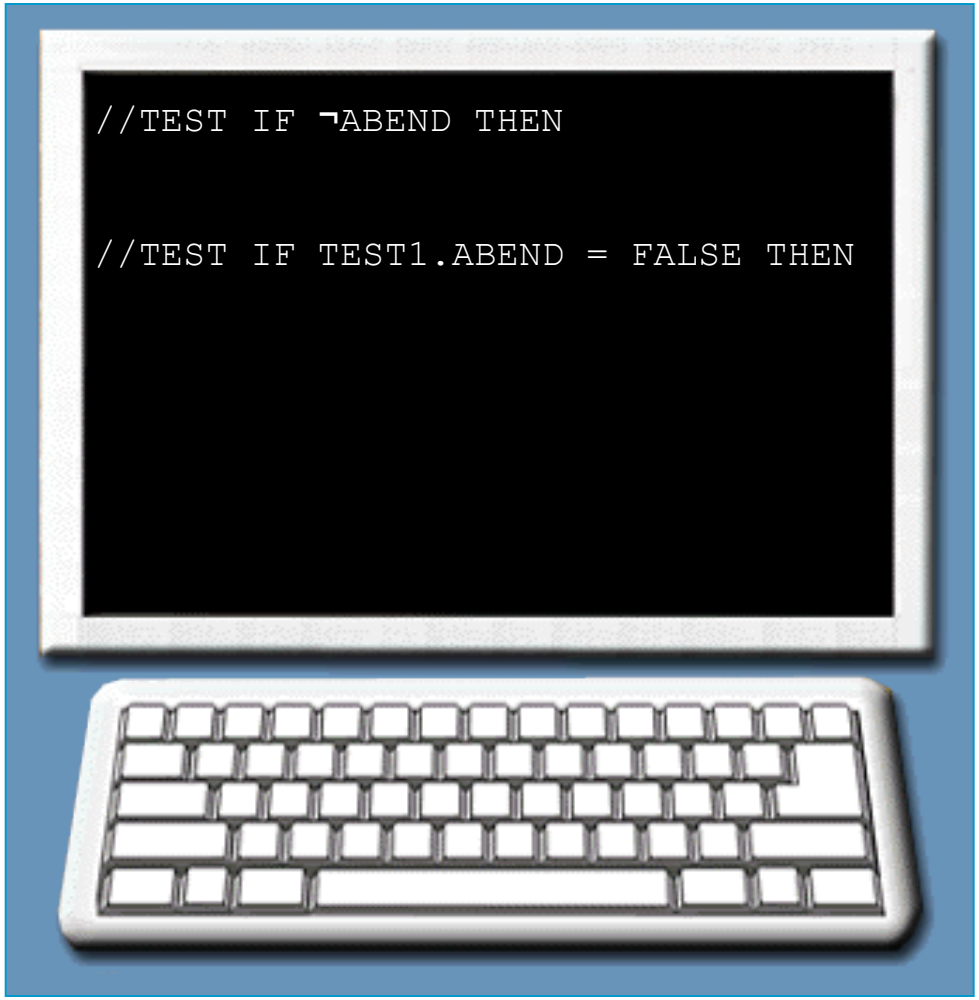
The keyword ¬ABEND checks for abnormal termination from any previous job step. The syntax used for ABEND is:

```
//name IF ¬ABEND THEN
```

or

```
//name IF ABEND = FALSE THEN
```

Both these statements will test to ensure an abnormal termination did not occur in any of the previous steps.



IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete following JCL statement using the ABEND operator to test for an abnormal termination in a prior job step named TEST1.

//TST4ABND IF (_____) THEN

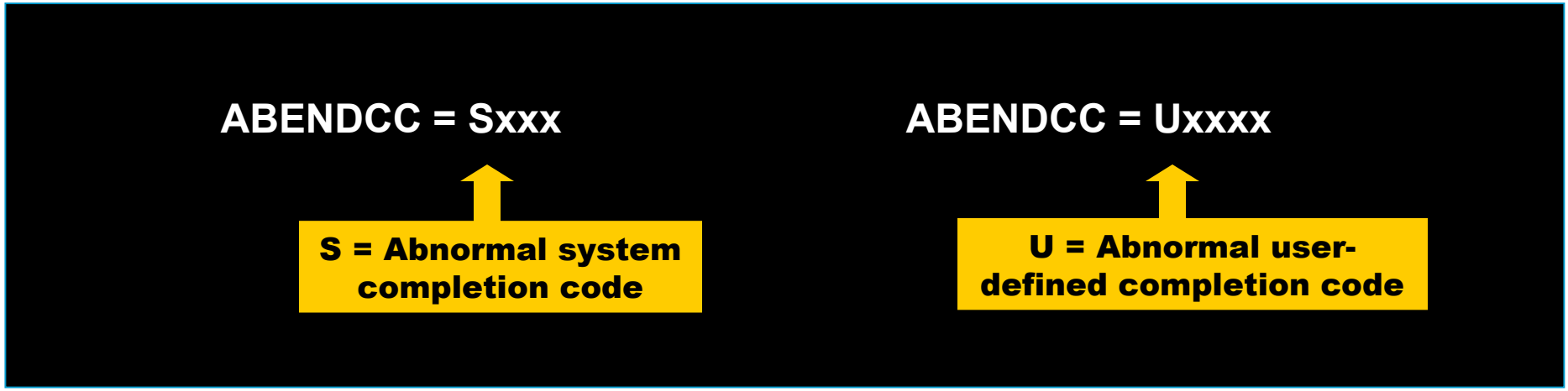
IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the ¬ABEND operator to test that an abnormal termination did not occur in a prior job step named TEST1.

//TST4ABND IF (_____) THEN

Relational-Expression keywords – ABENDCC.



What are the characteristics of the ABENDCC keyword?

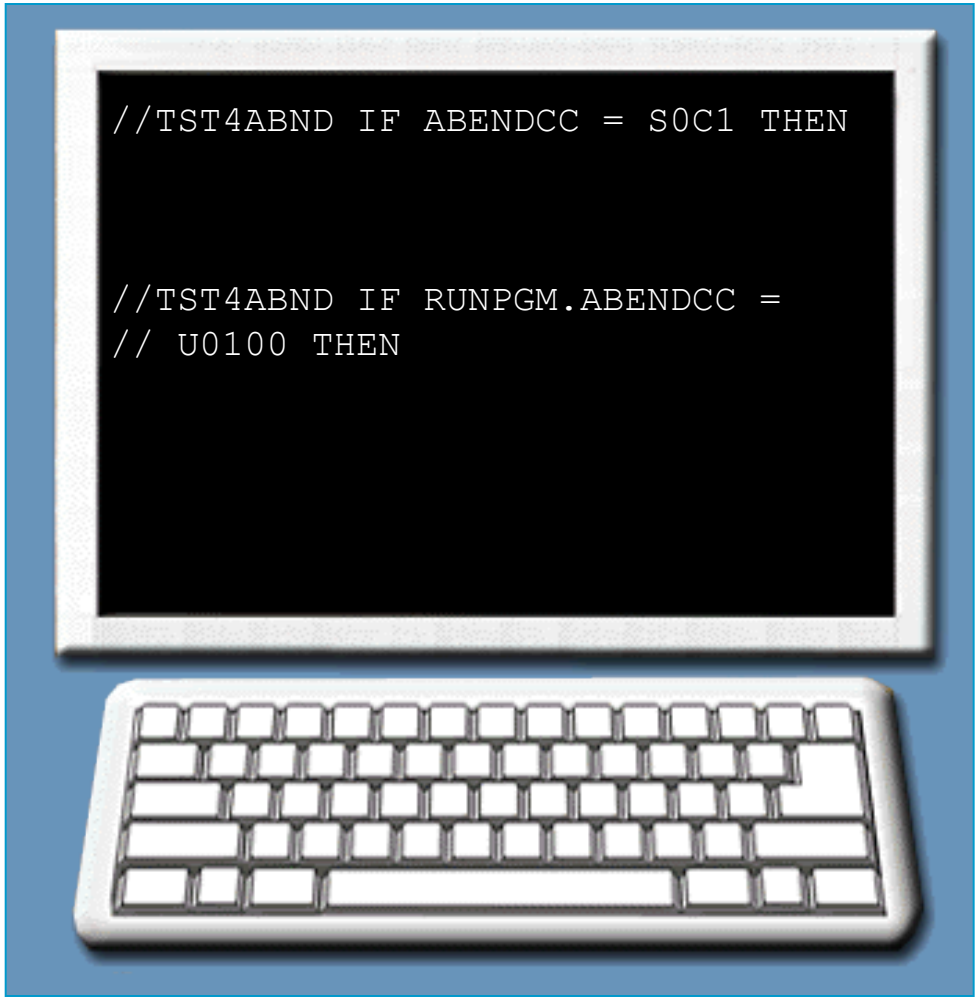
The relational-expression keyword ABENDCC tests for a specific system abend completion code or user defined abend completion code from any previous job step.

IF/THEN/ELSE/ENDIF construct.

ABENDCC – an example.

The first statement tests for an abnormal system completion code of 0C1 in the previous job step.

The second statement tests for an abnormal user-defined completion code of U0100 in a prior job step named RUNPGM in the previous job step.



Relational-Expression keywords – RUN.

What are the characteristics of the RUN keyword?

The keyword RUN tests to make sure that a specific job step or procedure step has been executed.

The syntax used for RUN is:

```
//name    IF stepname.RUN THEN
              or
//name    IF stepname.RUN = TRUE THEN
```

Relational-Expression keywords – ¬RUN.

What are the characteristics of the ¬RUN keyword?

The keyword ¬RUN tests to see if a specific job step or procedure step failed to execute.

The syntax used for ¬RUN is:

```
//name    IF ¬stepname.RUN THEN
              or
//name    IF stepname.RUN = FALSE THEN
```

Relational-Expression keywords – ¬RUN – an example.

In this example, the ¬RUN keyword tests if a step called LINK did not execute:

```
//JOB1      JOB 777,SMITH
//COMPILE   EXEC PGM=COMPILE
//LINK      EXEC PGM=LINK,
//            COND=(8,LT,COMPILE)
//TST4RUN   IF ¬LINK.RUN THEN
//TEST      EXEC PGM=RECOVER
```

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the RUN operator to test if a step named COMPILE executed successfully.

```
//TST4RUN   IF (_____________________) THEN
```

IF/THEN/ELSE/ENDIF construct.

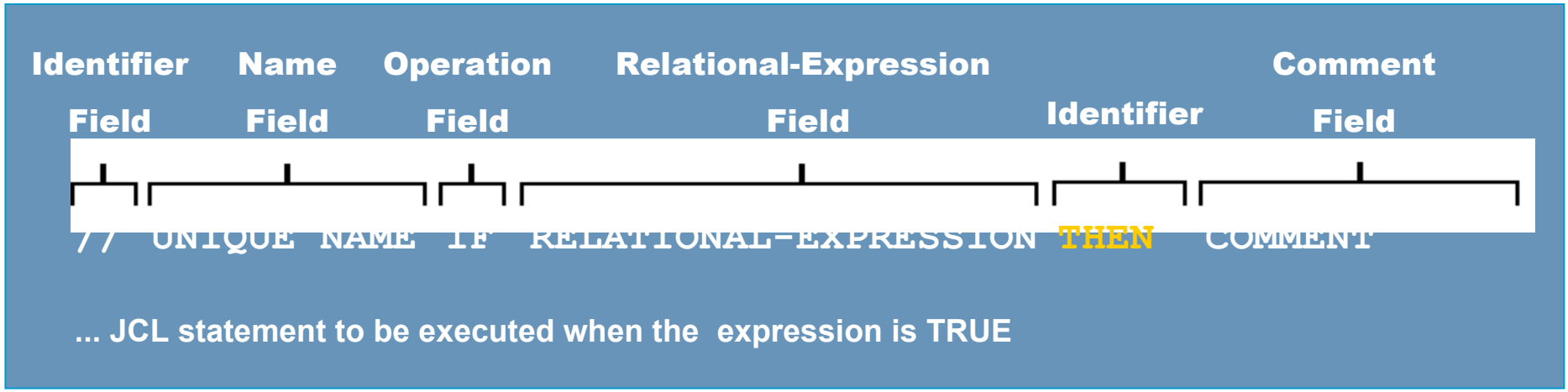
Are we on track?

Complete the following JCL statement using the ¬RUN operator to test if a step named COMPILE did not execute successfully.

//TST4RUN IF (_____) THEN

IF/THEN/ELSE/ENDIF construct.

The THEN clause.



What is the Then clause?

The THEN clause in an IF/THEN/ELSE/ENDIF statement construct contains the JCL statements that exist between the IF/THEN statement and either:

A corresponding ELSE statement (if one is specified) or
A corresponding ENDIF statement

The purpose of the THEN clause is to provide a route of execution if the condition specified in the IF statement tests true. If no JCL statements are specified, then the THEN clause becomes a null THEN clause.

The THEN clause – an example.

To illustrate the working of the THEN clause consider the following JCL code:

```
//TESTRC   IF (RC>=8) THEN
//ERROR    EXEC PGM=DELFILES
//   ENDIF
//STEP2     EXEC PGM=IEBCOPY
```

The THEN clause contains one JCL statement named ERROR. The program DELFILES, specified in the ERROR EXEC statement, will not execute unless the return code from any previous step is greater than or equal to 8.

Irrespective of the value of the return code, the program IEBCOPY specified in STEP2 will run as it is not part of the IF/THEN/ELSE/ENDIF statement construct.

IF/THEN/ELSE/ENDIF construct.

The ELSE clause.

What is the ELSE clause?

The ELSE clause in an IF/THEN/ELSE/ENDIF statement construct contains the JCL statements that exist between the ELSE keyword and a corresponding ENDIF statement.

The purpose of the ELSE clause is to provide a route of execution if the condition specified in the IF statement tests false. If no JCL statements are specified, then the ELSE clause becomes a null ELSE clause.

The ELSE clause – an example.

To illustrate the working of the ELSE consider the following JCL:

```
//TESTRUN   IF STEP1.RUN THEN
//GOOD       EXEC PGM=CREATE
//   ELSE
//ERROR      EXEC PGM=DELFILES
//   ENDIF
//STEP2      EXEC PGM=COMPRESS
```

The THEN contains one JCL statement named GOOD. The program CREATE, specified in the GOOD EXEC statement, will not be executed unless STEP1 has been executed successfully.

If STEP1 failed to execute, then a program DELFILES will be executed as it is contained under the ELSE.

Irrespective of whether STEP1 was executed successfully or not, the program COMPRESS specified in STEP2 will run as if it is not part of the IF/THEN/ELSE/ENDIF construct.

IF/THEN/ELSE/ENDIF construct.

The ENDIF clause.

What is the ENDIF clause?

To end an IF/THEN/ELSE/ENDIF statement construct an ENDIF clause must be coded.

One ENDIF clause is required for each IF statement coded.

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete following JCL statement up to THEN by checking if the return code is equal to 8 using parentheses.

```
//TST4RUN   IF _____________________
//GOOD       EXEC PGM=CREATE
```

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete following JCL statement using the ELSE clause including the identifier field.

```
//TST4RUN   IF (RC=8) THEN
//GOOD       EXEC PGM=CREATE

  _____
//ERROR      EXEC PGM=DELFILES
```

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete following JCL statement using the ENDIF clause including the identifier field.

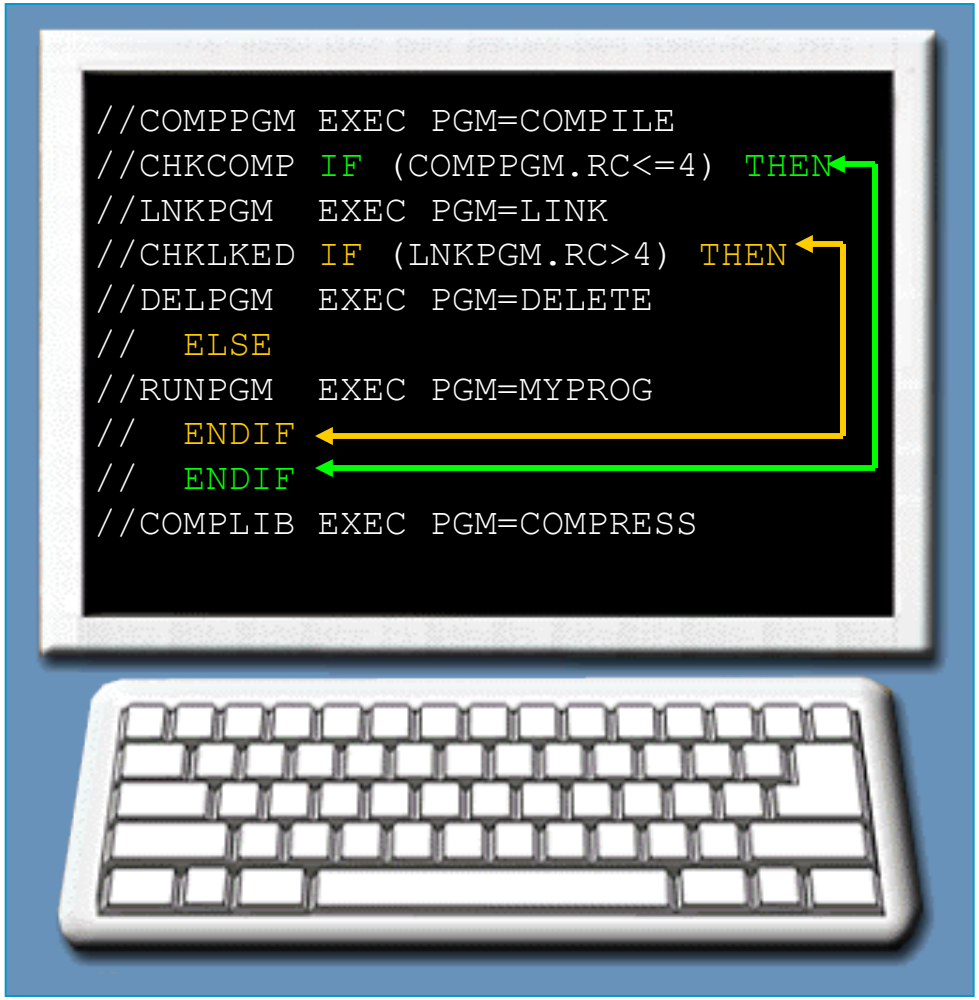
```
//TST4RUN   IF (RC=8) THEN
//GOOD       EXEC PGM=CREATE
//   ELSE
//ERROR      EXEC PGM=DELFILES
_____
//STEP2      EXEC PGM=COMPRESS
```

Nesting conditional constructs.

In a nested conditional construct the THEN clause or the ELSE clause (or both) will contain an additional IF/THEN/ELSE/ENDIF construct.

Each additional construct will have its own corresponding IF/THEN,ELSE and ENDIF statements.

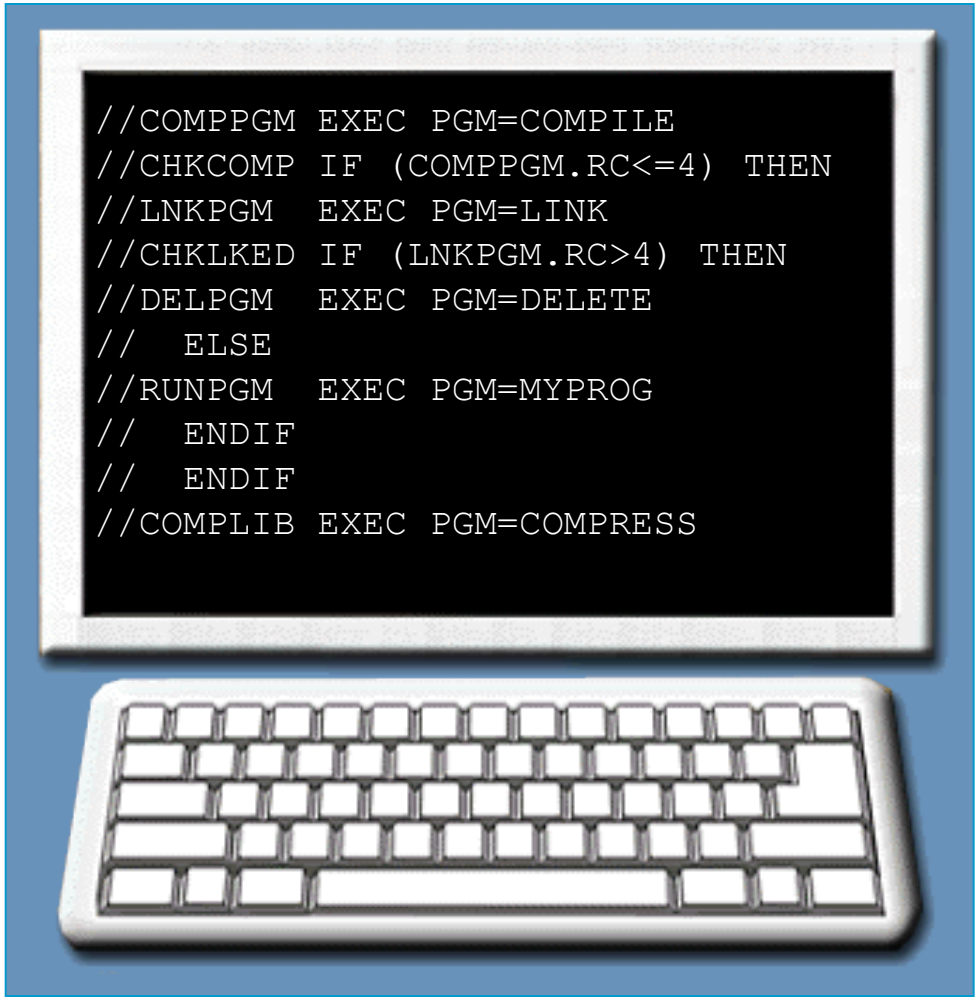
The example shows a nested conditional construct where the value of return code of a program determines the next step.



Nesting conditional constructs – an example.

In the outer IF/THEN/ELSE/ENDIF construct, the CHKCOMP statement checks to see if the return code from the step named COMPPGM is less than or equal to 4.

If the COMPPGM return code is less or equal to 4, then the LNKPGM step runs.

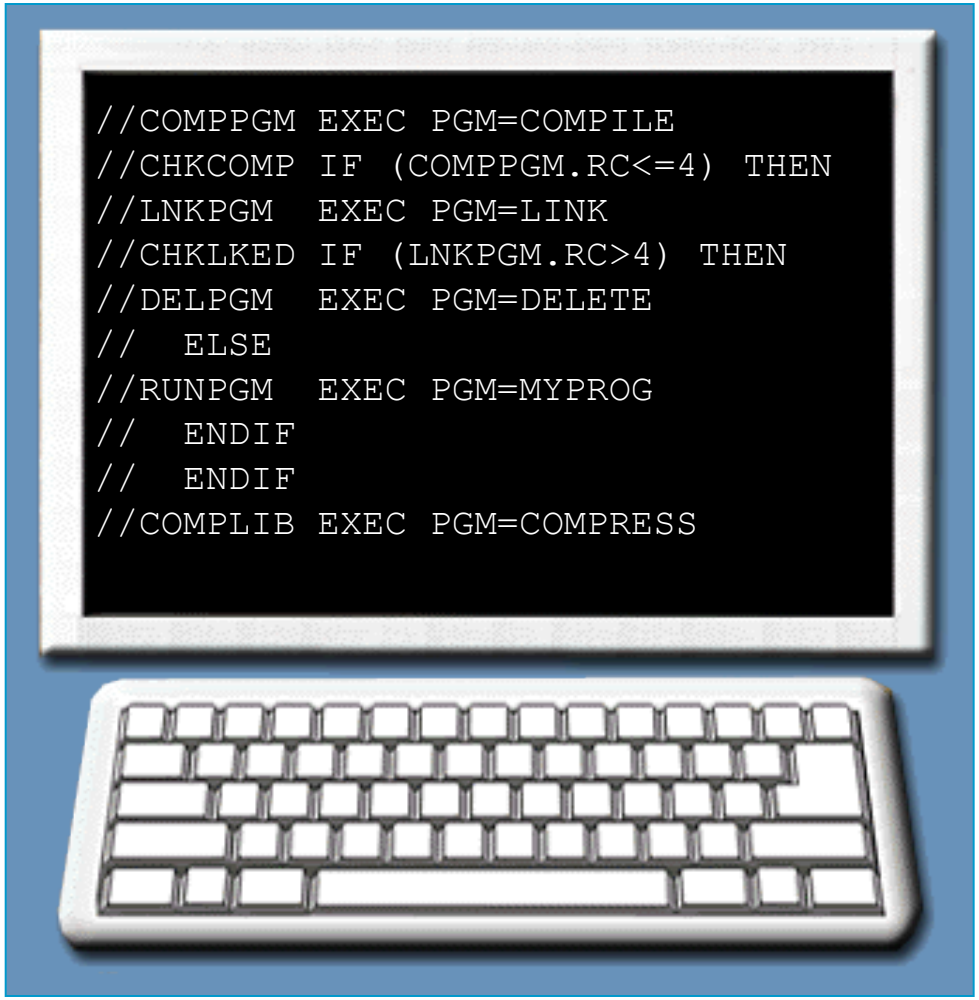


Nesting conditional constructs – an example.

Next the inner IF/THEN/ELSE/ENDIF construct is invoked, if the return code from LNKPGM step is greater than 4, the DELETE program (in the DELPGM step) executes.

If the return code from LNKPGM is less than or equal to 4, step RUNPGM will be run.

Step COMPLIB will be executed regardless of any conditional testing.



Nesting conditional constructs.

The COND parameter versus Conditional processing.

What is the advantage of using Conditional Processing over using the COND parameter?

Both the IF/THEN/ELSE/ENDIF statement construct and the COND parameter are used to conditionally control the execution of job steps.

It is always recommended to use the IF/THEN/ELSE/ENDIF statement construct for conditional testing as it is easier to read and understand compared to the COND parameter.

Unit summary.

Now that you have completed this unit, you should be able to:

- **Identify the various types of job conditions that an IF/THEN/ELSE/ENDIF statement construct can test.**
- **Code IF/THEN/ELSE/ENDIF statement constructs.**
- **Correct invalid IF/THEN, ELSE, and ENDIF JCL statements.**
- **State reasons why IF/THEN/ELSE/ENDIF JCL errors occur.**
- **State the advantages of using the IF/THEN/ELSE/ENDIF statement construct instead of the COND parameter.**

JCL

Chapter a6
Conditional processing

Chapter a6

Conditional processing

Conditional processing.

Unit introduction.

Be able to:

- **Steps in a job can be conditionally executed by using the IF/THEN/ELSE/ENDIF statement construct.**
- **The IF/THEN/ELSE/ENDIF statement construct allows the execution of job steps based on return codes, abend conditions, system completion codes from a previous job or procedure step.**
- **This construct provides greater functionality than the COND statement and is easier to use, read and understand.**

6

To keep the examples in this unit simple, the DD statements are not shown unless they are referenced directly.

Course objectives.

Be able to:

- **Identify the various types of job conditions that an IF/THEN/ELSE/ENDIF statement construct can test.**
- **Code IF/THEN/ELSE/ENDIF statement constructs.**
- **Correct invalid IF/THEN, ELSE, and ENDIF JCL statements.**
- **State reasons why IF/THEN/ELSE/ENDIF JCL errors occur.**
- **State the advantages of using the IF/THEN/ELSE/ENDIF statement construct instead of the COND parameter.**

IF/THEN/ELSE/ENDIF construct.

Syntax for the IF/THEN/ELSE/ENDIF statement construct.

What is the syntax for an IF/THEN/ELSE/ENDIF statement construct?

The syntax for coding an IF/THEN/ELSE/ENDIF statement construct is:

```
//name    IF     (relational-expression)   THEN
//name    JCL    statements to be executed when relational-expression is true
//name    ELSE
//name    JCL    statements to be executed when relational-expression is false
//name    ENDIF
```

The IF/THEN/ELSE/ENDIF statement construct can be coded anywhere in the job after the JOB statement.

8

The name field in the IF/THEN/ELSE/ENDIF statement construct is optional but if specified it must follow the coding rules for names in JCL statements.

The Name field coding rules.

Even though the name field is optional, if one is coded then it must follow the normal coding rules for names in JCL such as:

- The name must begin in position 3. If you do not code a name then leave the position blank.
- The name must be unique within the job.
- The first character of the name has to be alphabetical or national and it cannot be a number.
- The remaining characters can be alphanumeric or national.
- The name field must be followed by at least one space.

Valid JCL Names

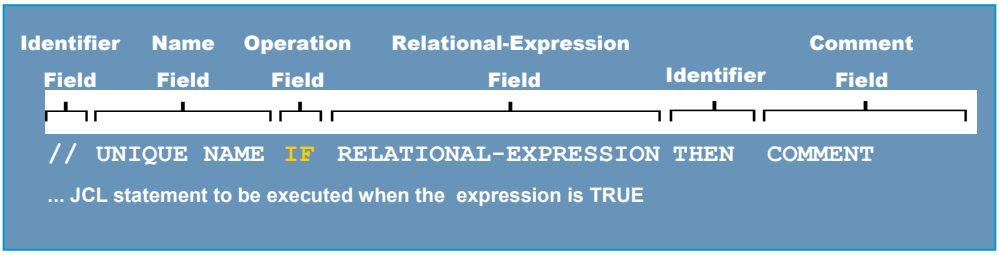
```
//CHECK1 IF (relational-expression) THEN
//UNIQUE IF (relational-expression) THEN
//PROG#1 IF (relational-expression) THEN
```

Invalid JCL Names

`// CHECK1`	(Does not begin in position 3)
`//TOOMANYCHARS`	(More than eight characters)
`//CHECK#2IF`	(Needs blank space after name)

IF/THEN/ELSE/ENDIF construct.

The Operation field – the IF statement.



What is the Operation Field?
The operation field contains the operators IF, ELSE, or ENDIF.

What are the characteristics of the IF statement?
The IF statement always precedes a relational-expression and the identifier THEN.

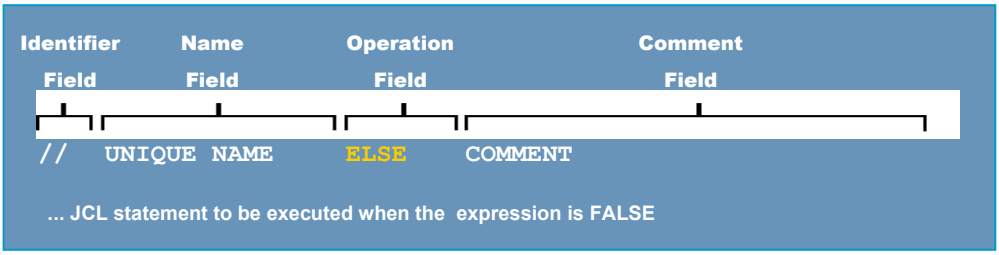
Following the IF statement are all of the JCL statements to be executed when the relational-expression is true. If there are none, then the IF statement should be followed immediately by the ELSE statement.

10

Anything coded after the THEN is considered a comment.

IF/THEN/ELSE/ENDIF construct.

The Operation field – The ELSE statement.



What are the characteristics of the ELSE statement?

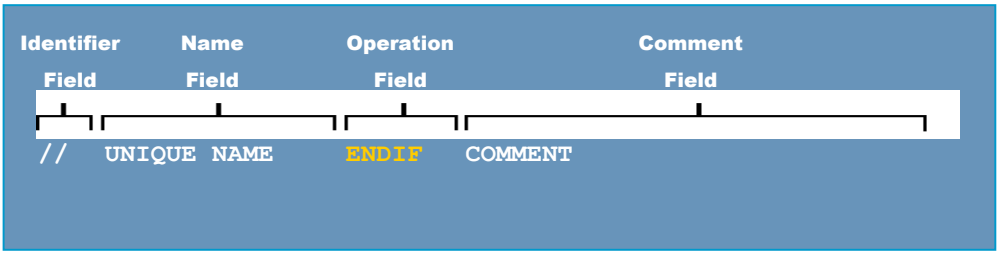
Following the ELSE statement are all the JCL statements to be executed when the relational-expression is false. If there are none, then the ELSE statement can be omitted.

The ELSE statement has no parameters.

Anything following the ELSE operator is considered a comment.

IF/THEN/ELSE/ENDIF construct.

The Operation field – the ENDIF statement.



What are the characteristics of the ENDIF statement?

The required ENDIF statement signifies the end of the IF/THEN/ELSE/ENDIF statement construct.

There must be at least one EXEC statement following either the IF statement or the ELSE statement.

Anything coded after the ENDIF statement is considered a comment by the operating system.

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Place the conditional statements in the proper order.

A. // ENDIF

B. //COND IF ABC>5 THEN

C. //STEP2 EXEC PGM=DELFILE

D. // ELSE

E. //STEP1 EXEC PROC=PRINT

13

The correct order is B., E., D., C., A.
(Assuming that STEP1 precedes STEP2.)

IF/THEN/ELSE/ENDIF construct.

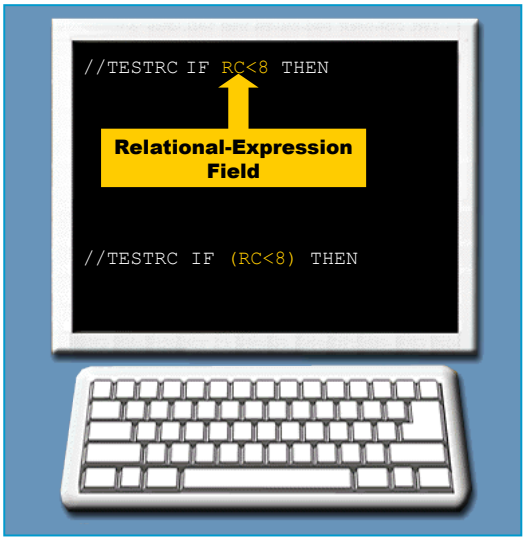
The Relational-Expression field.

What is the relational -expression field?

The relational-expression field follows the IF statement and specifies the condition that is evaluated at execution.

Depending on the values in the expression, the result of the condition is either true or false.

In the example, the first statement tests for a RC of less than 8. Hence the relational-expression is RC<8.



14

There must be at least one space between the IF operator and relational-expression field and similarly one space between the expression and the THEN operator.

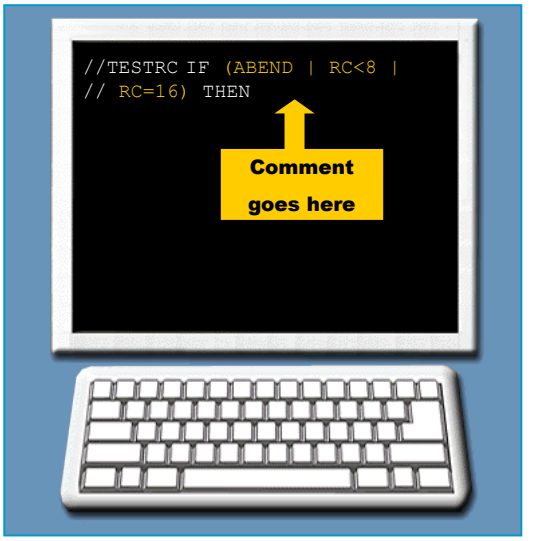
Also the relational-expression can be enclosed in parentheses, as shown in second statement.

IF/THEN/ELSE/ENDIF construct.

The Relational-Expression field.

Relational-expressions can be continued on more than one line.

To continue the expression, break the expression on a valid blank space and continue on the next line using columns 4 through 16.



15

Comments cannot be placed on the line you are continuing as the system will try to evaluate the comments as part of the expression.

IF/THEN/ELSE/ENDIF construct.

The Relational-Expression field.

A relational-expression can consist of any of the following, alone or in combination:

- **Comparison operators.**
- **Logical operators.**
- **NOT operators.**
- **Relational-expression keywords.**

16

The system evaluates comparison operators after any NOT operators and before any logical operators.

IF/THEN/ELSE/ENDIF construct.

Comparison operators.

What are the characteristics of a comparison operator?

Comparison operators compare a relational-expression keyword to a numeric value. The result of the comparison is either true or false.

The comparison operators are either alphabetic or arithmetic.

Operator	Meaning
GT or >	Greater than
GE or >=	Greater than or equal to
NG or ¬>	Not greater than
EQ or =	Equal to
NE or ¬=	Not equal to
LT or <	Less than
LE or <=	Less than or equal to
NL or ¬<	Not less than

IF/THEN/ELSE/ENDIF construct.

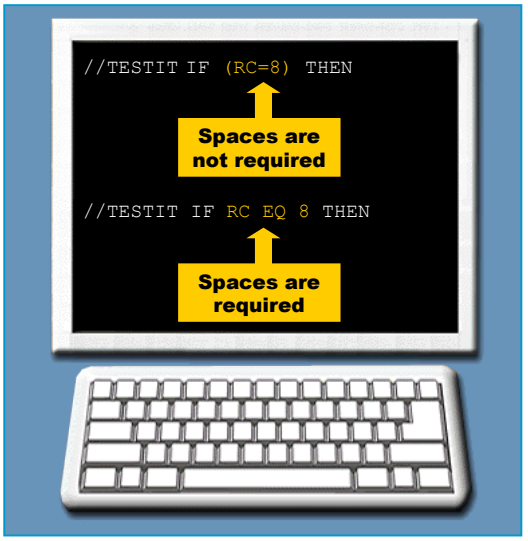
Comparison operators – an example.

In this example, the statement tests if a RC is equal to 8.

The relational-expression (RC=8) must be both preceded and followed by at least one space.

No spaces are required before or after an arithmetic operator, such as = or >.

At least one space is required both before and after alphabetic comparison operators, such as EQ or GT.



The use of parentheses in the relational-expression is optional, but it is useful when coding combinations of expressions.

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the JCL statement up to THEN operator to check if the return code is equal to 8, using parentheses.

//TESTIT _____

19

The correct answer is: IF (RC=8) THEN

IF/THEN/ELSE/ENDIF construct.

Logical operators.

The logical operators include:

AND (&)

OR (|)

NOT (¬)

NOT sign representation: CNTRL - ž

IF/THEN/ELSE/ENDIF construct.

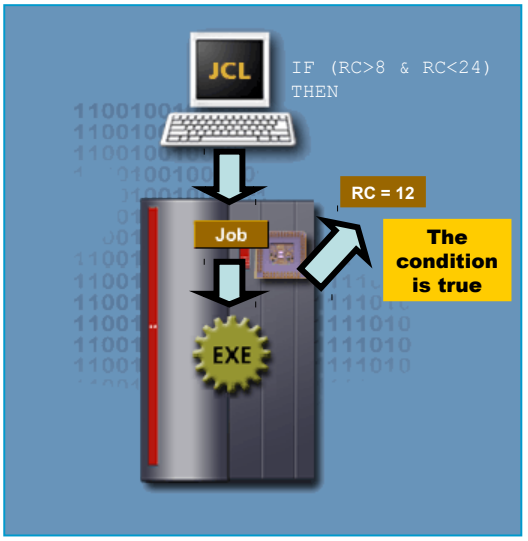
Logical operators - AND.

What are the characteristics of the AND operator?

The AND (&) operator returns a true value only if both relational -expressions are true.

For example, to test if a return code is between 8 and 24:

```
//TEST1 IF (RC>8 & RC<24) THEN
```



21

The AND operator must be preceded and followed by at least one space.

IF/THEN/ELSE/ENDIF construct.

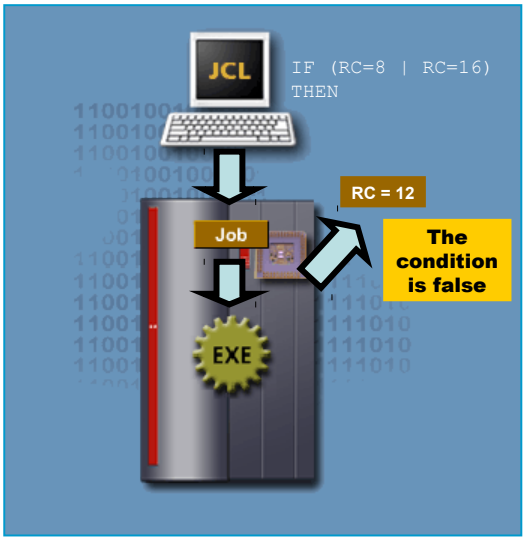
Logical operators - OR.

What are the characteristics of the OR operator?

The OR (|) operator returns a true value if either of the relational-expression is true.

For example, to test if a return code is either equal to 8 or 16:

```
//TEST2 IF (RC=8 | RC=16) THEN
```



22

The OR operator must be preceded and followed by at least one space.

IF/THEN/ELSE/ENDIF construct.

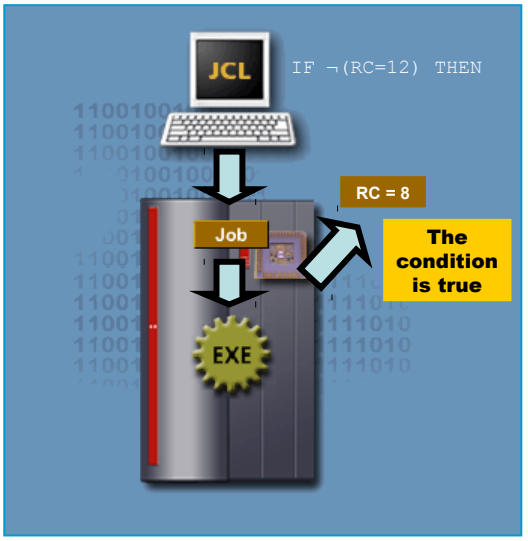
Logical operators - NOT.

What are the characteristics of the NOT operator?

The NOT (¬) operator reverses the testing of a relational-expression. The system evaluates the NOT operator before any comparisons or logical operators.

For example, to test if a return code is not equal to 12:

```
//TEST3 IF ¬(RC=12) THEN
```



23

The NOT operator does not require a space to separate it from the expression it is reversing.

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the AND (&) operator to test if the return code is between 8 and 24.

```
//TEST1    IF (____________________) THEN
```

24

The correct answer is: RC >= 8 AND RC <= 24

It means in the 8 – 24 range.

See ‘MCOE.EDU.JCL.JCL(IFAND)’

Try to test values 7, 8, 9 and 23, 24, 25.

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the OR (|) operator to test if the return code is equal to 8 or 16.

```
//TEST2    IF (____________________) THEN
```

25

The correct answer is: RC = 8 OR RC = 16

It means 8 or 16.

See 'MCOE.EDU.JCL.JCL(IFOR)'

Try to test values 7, 8, 9 and 15, 16, 17.

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the NOT (¬) operator (with parentheses) to test if the return code is not greater than 8 and not less than 24.

```
//TEST3    IF _____________________ THEN
```

26

The correct answer is: NOT ((RC > 8) AND (RC < 24))

It means every number from 0 to 8 and for 24 to infinity.

See 'MCOE.EDU.JCL.JCL(IFNOT)'

Try to test values 7, 8, 9 and 23, 24, 25.

Warning: specify condition even in open English as exact as possible.

Relational-Expression keywords.

What are the characteristics of relational-expression keywords?

Relational-expression keywords are used to test a RC, abend condition or abend completion code, or to test if a step began executing. The relational-expression keywords are:

- **RC**
- **ABEND**
- **¬ABEND**
- **ABENDCC**
- **RUN**
- **¬RUN**

Preceding the keyword with a step name relates the expression to a specific job step.
Syntax: **stepname.keyword**

Preceding the keyword with both a step name and procedure step name relates the expression to a specific procedure step.
Syntax: **stepname.procstepname.keyword**

IF/THEN/ELSE/ENDIF construct.

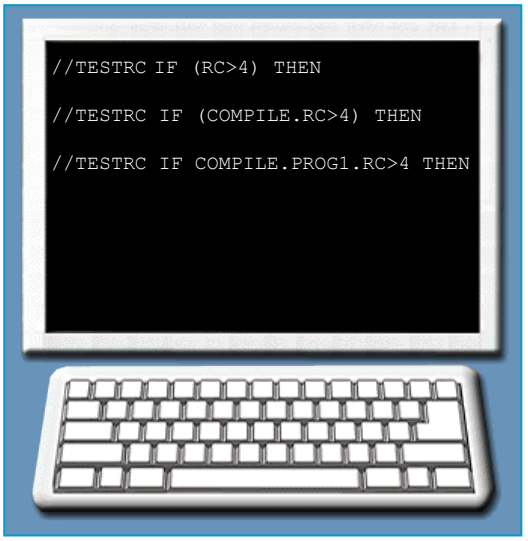
Relational-Expression keywords - RC.

RC represents the highest return code received from a previous job step.

In the example, the first statement checks if the previous job step had a return code > 4.

The second statement tests if a prior job step named COMPILE produced a return code > 4.

The third one checks if a specific procedure step, PROG1 in the job step COMPILE, produced a return code > 4.



IF/THEN/ELSE/ENDIF construct.

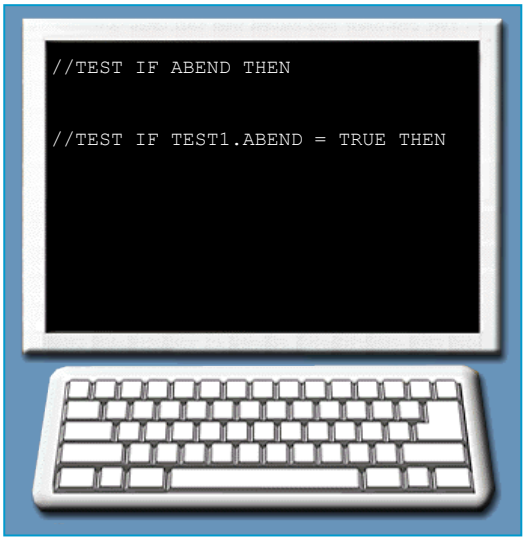
Relational-Expression keywords – ABEND.

What are the characteristics of the ABEND keyword?

The keyword ABEND tests for abnormal termination from any previous job step. The syntax used for ABEND is:

```
//name IF ABEND THEN
        or
//name IF ABEND = TRUE THEN
```

Both these statements will test for an abnormal termination in any of the previous steps.



29

To check a specific job step for abnormal termination:

stepname.ABEND

or

stepname.ABEND = TRUE

To check a specific procedure step for abnormal termination:

stepname.procstepname.ABEND

or

stepname.procstepname.ABEND = TRUE

Both formats can be preceded with a step name or procedure step name to check specific job steps or procedure steps for abnormal termination.

IF/THEN/ELSE/ENDIF construct.

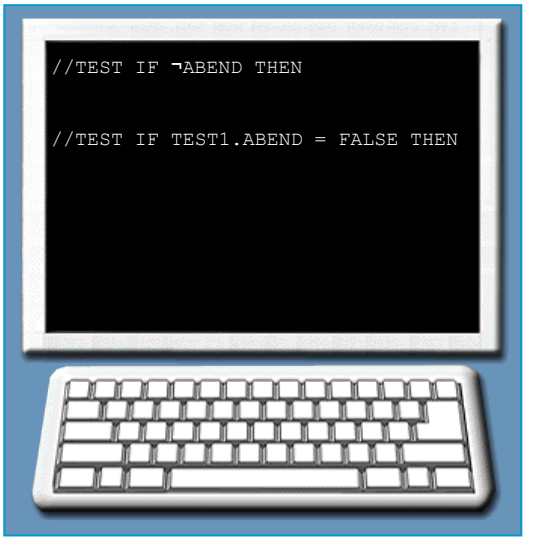
Relational-Expression keywords - ¬ABEND.

What are the characteristics of the ¬ABEND keyword?

The keyword ¬ABEND checks for abnormal termination from any previous job step. The syntax used for ABEND is:

```
//name IF ¬ABEND THEN
            or
//name IF ABEND = FALSE THEN
```

Both these statements will test to ensure an abnormal termination did not occur in any of the previous steps.



30

To check that a specific step did not result in an abnormal termination:

¬stepname.ABEND

or

stepname.ABEND = FALSE

To check that a specific procedure step did not result in an abnormal termination:

¬stepname.procstepname.ABEND

or

stepname.procstepname.ABEND = FALSE

Both formats can be preceded with a step name or procedure step name to ensure that abnormal termination did not occur in specific job steps or procedure steps.

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete following JCL statement using the ABEND operator to test for an abnormal termination in a prior job step named TEST1.

//TST4ABND IF (_____) THEN

31

The correct answer is: TEST1.ABEND

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the ¬ABEND operator to test that an abnormal termination did not occur in a prior job step named TEST1.

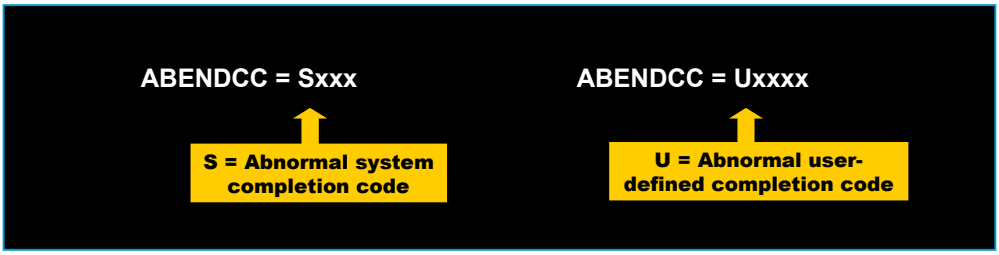
//TST4ABND IF (_____) THEN

32

The correct answer is: ¬TEST1.ABEND

IF/THEN/ELSE/ENDIF construct.

Relational-Expression keywords – ABENDCC.



What are the characteristics of the ABENDCC keyword?

The relational-expression keyword ABENDCC tests for a specific system abend completion code or user defined abend completion code from any previous job step.

33

The character S in the first expression indicates an abnormal system completion code, and the xxx represents the three digit hexadecimal abend code.

The U in the second expression indicates an abnormal user-defined completion code, and the xxxx represents the four digit hexadecimal abend code.

To test the abend code from a specific step

stepname.ABENDCC = Sxxx

stepname.ABENDCC = Uxxxx

To test the abend code from a specific procedure step

stepname.procstepname.ABENDCC = Sxxx

stepname.procstepname.ABENDCC = Uxxxx

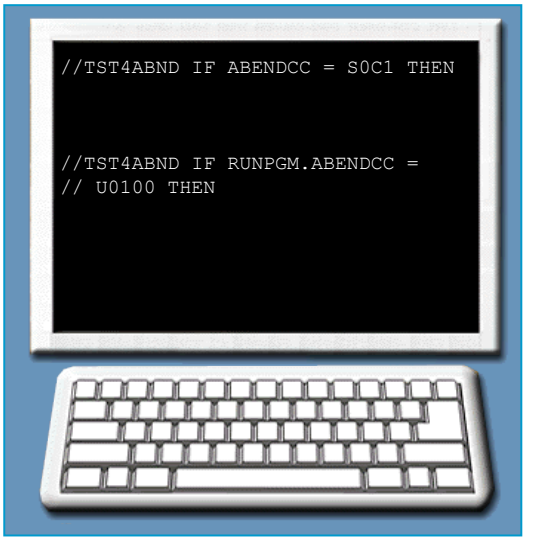
Both formats can be preceded with a step name or procedure step name to test the abend code from specific job steps or procedure steps.

IF/THEN/ELSE/ENDIF construct.

ABENDCC – an example.

The first statement tests for an abnormal system completion code of 0C1 in the previous job step.

The second statement tests for an abnormal user-defined completion code of U0100 in a prior job step named RUNPGM in the previous job step.



IF/THEN/ELSE/ENDIF construct.

Relational-Expression keywords – RUN.

What are the characteristics of the RUN keyword?

The keyword RUN tests to make sure that a specific job step or procedure step has been executed.

The syntax used for RUN is:

```
//name   IF stepname.RUN THEN
             or
//name   IF stepname.RUN = TRUE THEN
```

35

To test if a specific procedure step has been executed:

//name IF stepname.procstepname.RUN THEN

A step name or both step name and procedure step name must precede the RUN keyword.

IF/THEN/ELSE/ENDIF construct.

Relational-Expression keywords – ¬RUN.

What are the characteristics of the ¬RUN keyword?

The keyword ¬RUN tests to see if a specific job step or procedure step failed to execute.

The syntax used for ¬RUN is:

```
//name   IF ¬stepname.RUN THEN
             or
//name   IF stepname.RUN = FALSE THEN
```

36

To test if a specific procedure step failed to execute:

//name IF ¬stepname.procstepname.RUN THEN

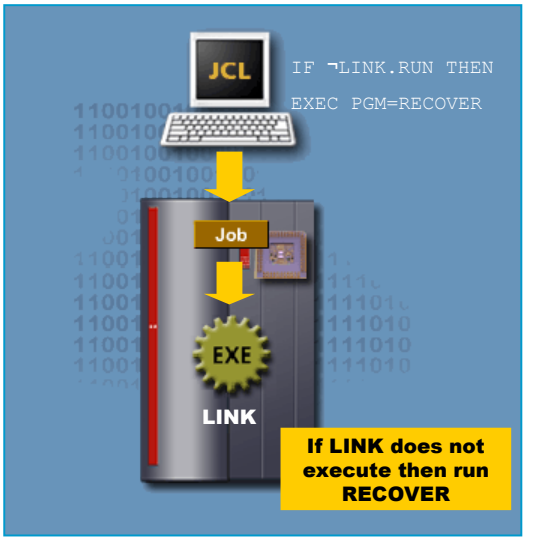
A step name or both step name and procedure step name must precede the ¬RUN keyword.

IF/THEN/ELSE/ENDIF construct.

Relational-Expression keywords – ¬RUN – an example.

In this example, the ¬RUN keyword tests if a step called LINK did not execute:

```
//JOB1      JOB 777,SMITH
//COMPILE   EXEC PGM=COMPILE
//LINK      EXEC PGM=LINK,
//          COND=(8,LT,COMPILE)
//TST4RUN   IF ¬LINK.RUN THEN
//TEST      EXEC PGM=RECOVER
```



IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the RUN operator to test if a step named COMPILE executed successfully.

```
//TST4RUN   IF (____________________) THEN
```

38

The correct answer is: COMPILE.RUN

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete the following JCL statement using the ¬RUN operator to test if a step named COMPILE did not execute successfully.

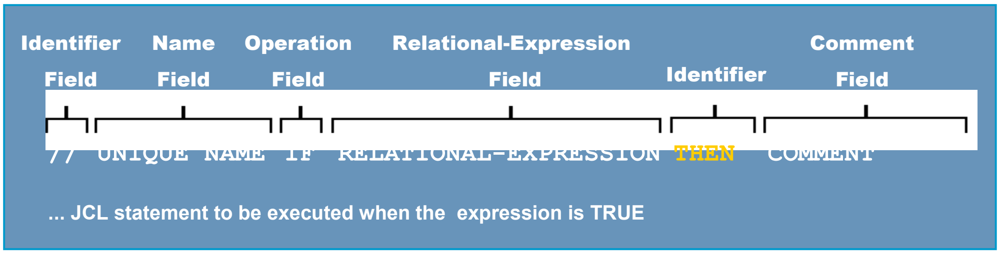
```
//TST4RUN    IF (_____________________) THEN
```

39

The correct answer is: ¬COMPILE.RUN

IF/THEN/ELSE/ENDIF construct.

The THEN clause.



What is the Then clause?

The THEN clause in an IF/THEN/ELSE/ENDIF statement construct contains the JCL statements that exist between the IF/THEN statement and either:

A corresponding ELSE statement (if one is specified) or
A corresponding ENDIF statement

The purpose of the THEN clause is to provide a route of execution if the condition specified in the IF statement tests true. If no JCL statements are specified, then the THEN clause becomes a null THEN clause.

The THEN clause – an example.

To illustrate the working of the THEN clause consider the following JCL code:

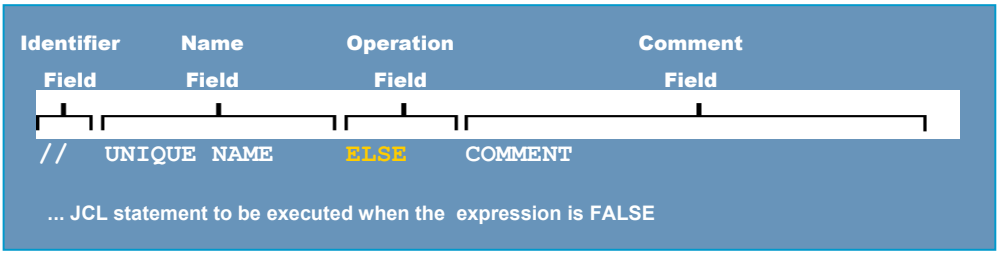
```
//TESTRC    IF (RC>=8) THEN
//ERROR     EXEC PGM=DELFILES
//   ENDIF
//STEP2      EXEC PGM=IEBCOPY
```

The THEN clause contains one JCL statement named ERROR. The program DELFILES, specified in the ERROR EXEC statement, will not execute unless the return code from any previous step is greater than or equal to 8.

Irrespective of the value of the return code, the program IEBCOPY specified in STEP2 will run as it is not part of the IF/THEN/ELSE/ENDIF statement construct.

IF/THEN/ELSE/ENDIF construct.

The ELSE clause.



What is the ELSE clause?

The ELSE clause in an IF/THEN/ELSE/ENDIF statement construct contains the JCL statements that exist between the ELSE keyword and a corresponding ENDIF statement.

The purpose of the ELSE clause is to provide a route of execution if the condition specified in the IF statement tests false. If no JCL statements are specified, then the ELSE clause becomes a null ELSE clause.

The ELSE clause – an example.

To illustrate the working of the ELSE consider the following JCL:

```
//TESTRUN   IF STEP1.RUN THEN
//GOOD       EXEC PGM=CREATE
//   ELSE
//ERROR      EXEC PGM=DELFILES
//   ENDIF
//STEP2      EXEC PGM=COMPRESS
```

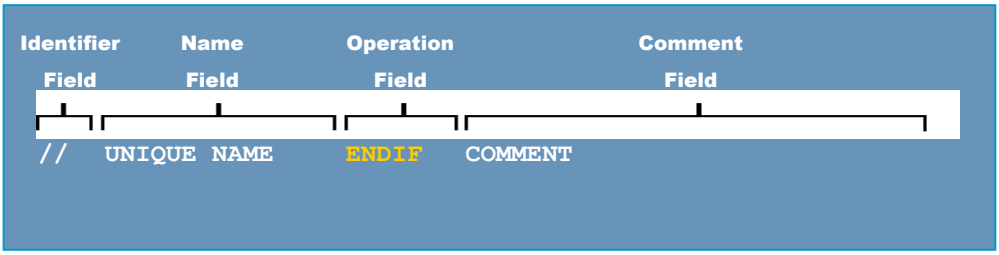
The THEN contains one JCL statement named GOOD. The program CREATE, specified in the GOOD EXEC statement, will not be executed unless STEP1 has been executed successfully.

If STEP1 failed to execute, then a program DELFILES will be executed as it is contained under the ELSE.

Irrespective of whether STEP1 was executed successfully or not, the program COMPRESS specified in STEP2 will run as if it is not part of the IF/THEN/ELSE/ENDIF construct.

IF/THEN/ELSE/ENDIF construct.

The ENDIF clause.



What is the ENDIF clause?

To end an IF/THEN/ELSE/ENDIF statement construct an ENDIF clause must be coded.

One ENDIF clause is required for each IF statement coded.

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete following JCL statement up to THEN by checking if the return code is equal to 8 using parentheses.

```
//TST4RUN   IF _____________________
//GOOD        EXEC PGM=CREATE
```

45

The correct answer is: (RC=8) THEN

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete following JCL statement using the ELSE clause including the identifier field.

```
//TST4RUN   IF (RC=8) THEN
//GOOD      EXEC PGM=CREATE
  ___________
//ERROR     EXEC PGM=DELFILES
```

46

The correct answer is: // ELSE

IF/THEN/ELSE/ENDIF construct.

Are we on track?

Complete following JCL statement using the ENDIF clause including the identifier field.

```
//TST4RUN   IF (RC=8) THEN
//GOOD       EXEC PGM=CREATE
//   ELSE
//ERROR      EXEC PGM=DELFILES
_____
//STEP2      EXEC PGM=COMPRESS
```

47

The correct answer is: // ENDIF

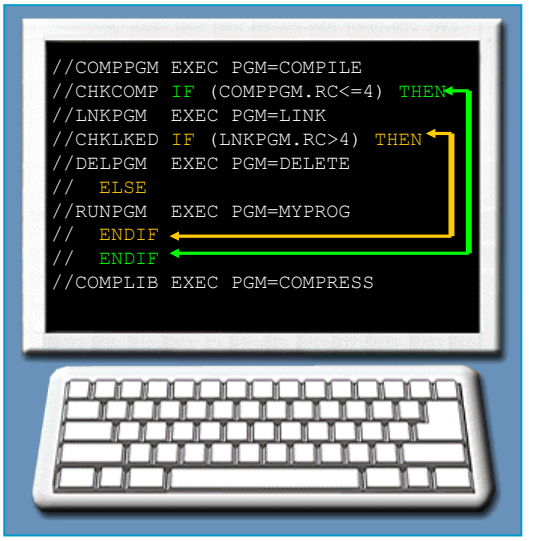
Nesting conditional constructs.

Nesting conditional constructs.

In a nested conditional construct the THEN clause or the ELSE clause (or both) will contain an additional IF/THEN/ELSE/ENDIF construct.

Each additional construct will have its own corresponding IF/THEN,ELSE and ENDIF statements.

The example shows a nested conditional construct where the value of return code of a program determines the next step.



48

The IF/THEN/ELSE/ENDIF statement construct can be nested up to 15 levels.

Nesting conditional constructs.

Nesting conditional constructs – an example.

In the outer IF/THEN/ELSE/ENDIF construct, the CHKCOMP statement checks to see if the return code from the step named COMPPGM is less than or equal to 4.

If the COMPPGM return code is less or equal to 4, then the LNKPGM step runs.

49

See the next page...

Nesting conditional constructs.

Nesting conditional constructs – an example.

Next the inner IF/THEN/ELSE/ENDIF construct is invoked, if the return code from LNKPGM step is greater than 4, the DELETE program (in the DELPGM step) executes.

If the return code from LNKPGM is less than or equal to 4, step RUNPGM will be run.

Step COMPLIB will be executed regardless of any conditional testing.

Nesting conditional constructs.

The COND parameter versus Conditional processing.

What is the advantage of using Conditional Processing over using the COND parameter?

Both the IF/THEN/ELSE/ENDIF statement construct and the COND parameter are used to conditionally control the execution of job steps.

It is always recommended to use the IF/THEN/ELSE/ENDIF statement construct for conditional testing as it is easier to read and understand compared to the COND parameter.

Nesting conditional constructs.

Unit summary.

Now that you have completed this unit, you should be able to:

- **Identify the various types of job conditions that an IF/THEN/ELSE/ENDIF statement construct can test.**
- **Code IF/THEN/ELSE/ENDIF statement constructs.**
- **Correct invalid IF/THEN, ELSE, and ENDIF JCL statements.**
- **State reasons why IF/THEN/ELSE/ENDIF JCL errors occur.**
- **State the advantages of using the IF/THEN/ELSE/ENDIF statement construct instead of the COND parameter.**