



JCL

Chapter a2

Coding JOB statements

Job Control Language

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

Job Control Language

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

Job Control Language

Chapter c1. Nested procedures

Chapter c2. Cataloging procedures

Chapter c3. Using utility programs

Chapter c4. Sample utility application

Chapter a2

Coding JOB statements

Coding JOB statements

Course objectives.

Be able to:

- **Explain the purpose and syntax of the JOB statement.**
- **Define the JOB name.**
- **Code positional parameters and keyword parameters.**
- **Code the most commonly used keyword parameters on a JOB statement.**
- **Complete the JOB statement.**
- **Code additional JOB statement parameters.**

The JOB statement

Defining a JOB statement.

A JOB statement is the first statement in any JCL code.

It marks the start of a job and gives the name of the job.

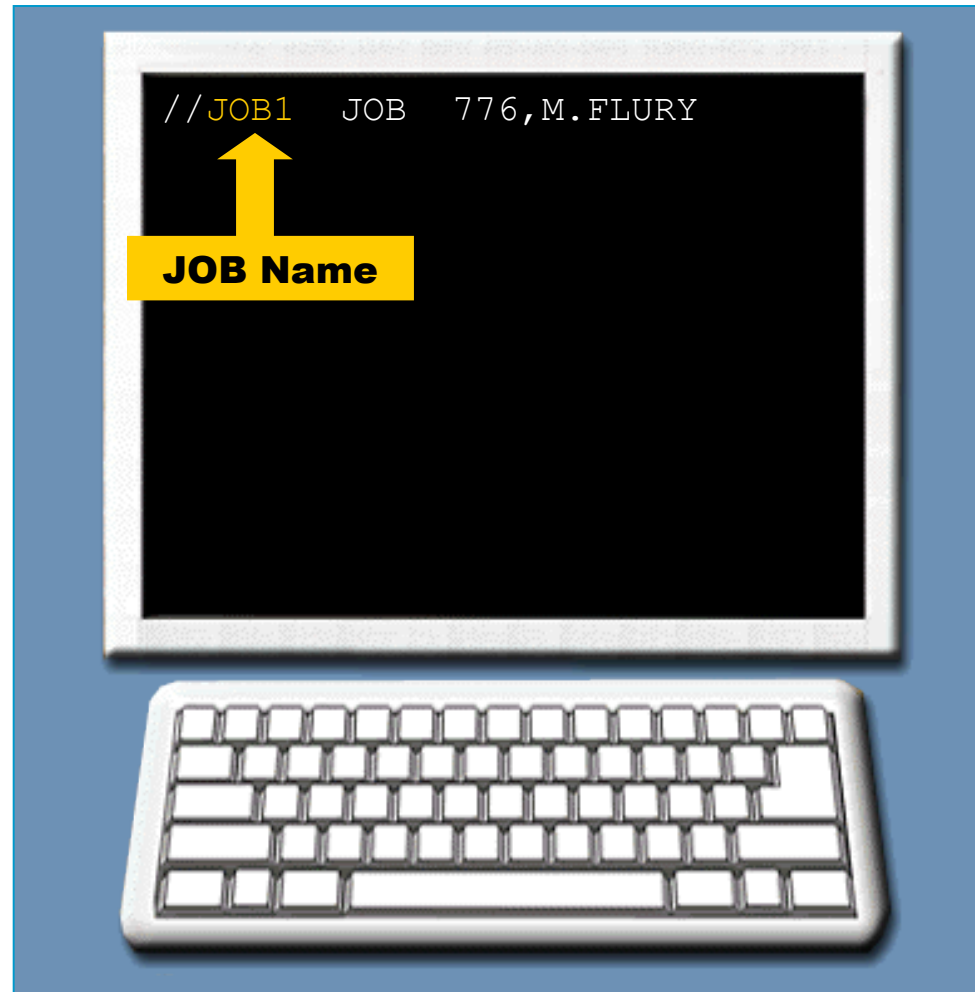
The parameters included in the JOB statement, such as accounting information and condition settings for job termination, apply to the entire job.

The JOB statement

The JOB name.

The JOB name is a 1 to 8 character name that identifies the job so that other JCL statements or the operating system can refer to it.

The JOB name must begin in position 3 with no spaces between it and the identifier.



The JOB statement.

Are we on track?

Code the JOB name as JOB1 including the identifier field and add the correct operation field.

_____ 776,M.FLURY

The JOB statement.

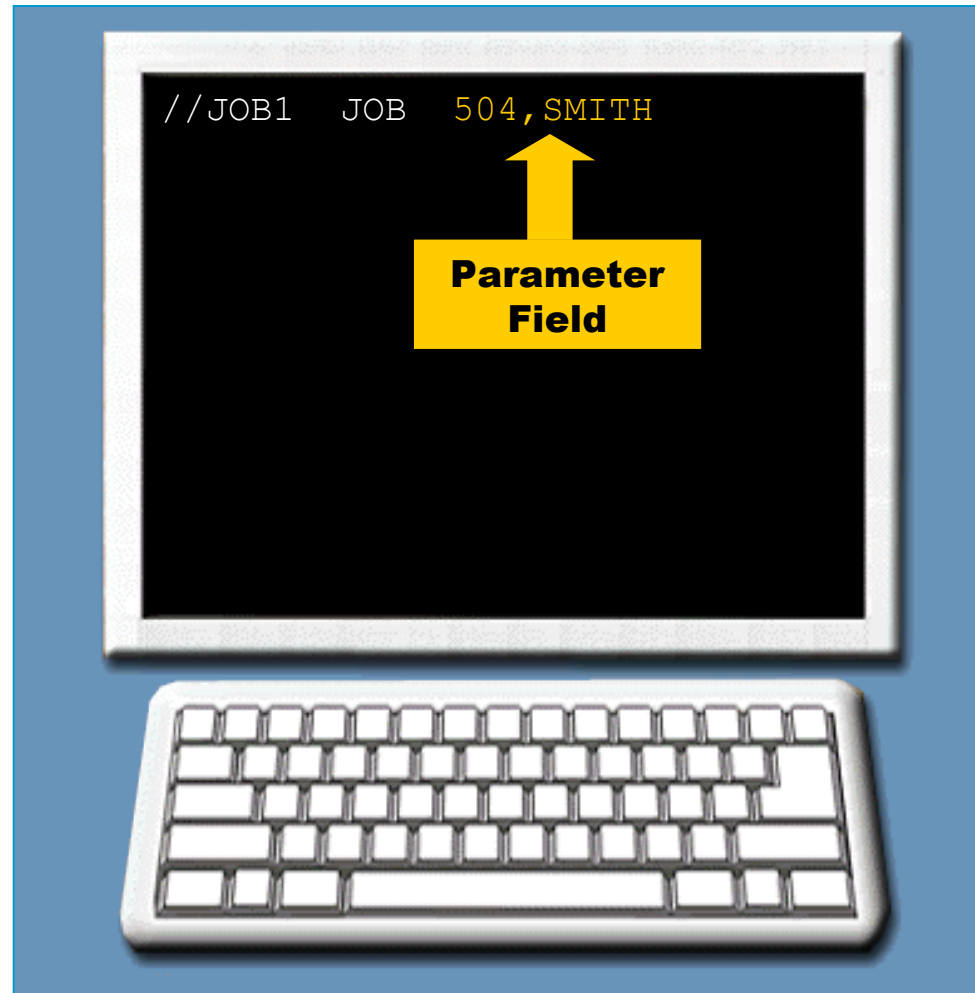
Parameter fields.

The parameter field, in the JOB statement defines information that applies to the entire job.

This information includes accounting number, programmer name, and additional information regarding the job.

One or more spaces separate the job parameters from the JOB operation field.

Always use a comma to separate parameters in a JOB statement.



The JOB statement.

Are we on track?

Code the parameter field with job accounting information as 776 and the programmer's name as M. Flury.

//JOB1 JOB _____

The JOB name

Nature of a job name.

The job name is the second field in a JOB statement. It follows the identifier field (//).

A JCL programmer should select the mandatory job name to identify the job to the operating system.

The operating system will not run jobs having the same name concurrently. Therefore, it is important that each job should be assigned a unique name.

If two jobs having the same name try to execute at the same time, the second one gets delayed till the first one completes.

The JOB name

Rules for job names: a summary.

Rules require a job name to begin in position 3 and to be 1 to 8 characters in length.

The first character of a job name should be either alphabetic or a national symbol. It should not be a number.

The rest of the characters in the job name can be either alphanumeric or they can be national symbols.

Special characters or spaces are not allowed.

Valid Job Names

//JOB1 JOB

//EXAMPLE4 JOB

//RUN#2 JOB

Invalid Job Names

//JOB1+ JOB (Includes a special character)

//EXAMPLE14 JOB (More than eight characters)

// RUN#2 JOB (Does not begin in position 3)

The JOB name.

Are we on track?

Place the following parts of a JOB statement in the correct order.

A. JOB

B. //

C. JOBNAME

D. PROGRAMMER

E. ACCOUNT

The JOB parameter field.

Positional parameters.

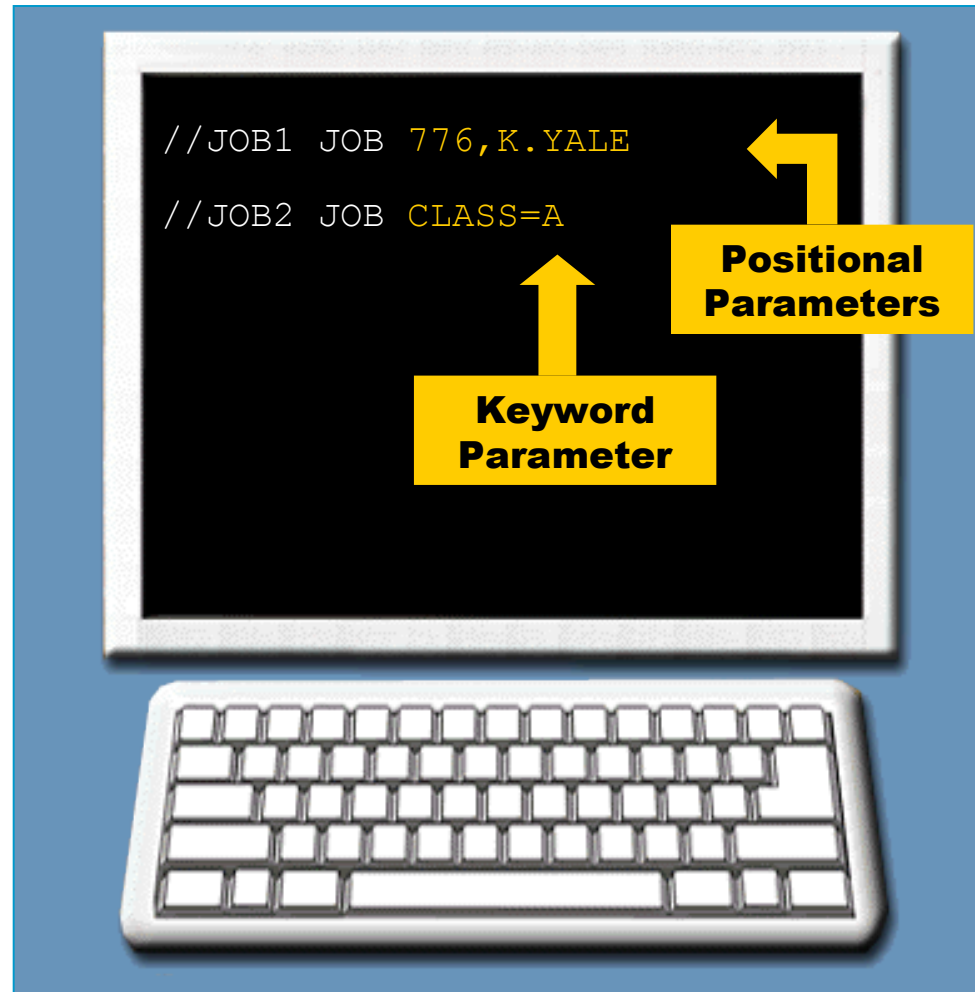
The parameter field of a JOB statement appears after the JOB operator field.

There are two types of parameters:

- Positional parameters.
- Keyword parameters.

What are positional parameters?

Positional parameters are characterized by their location in the parameter field in relation to other parameters.



The JOB parameter field.

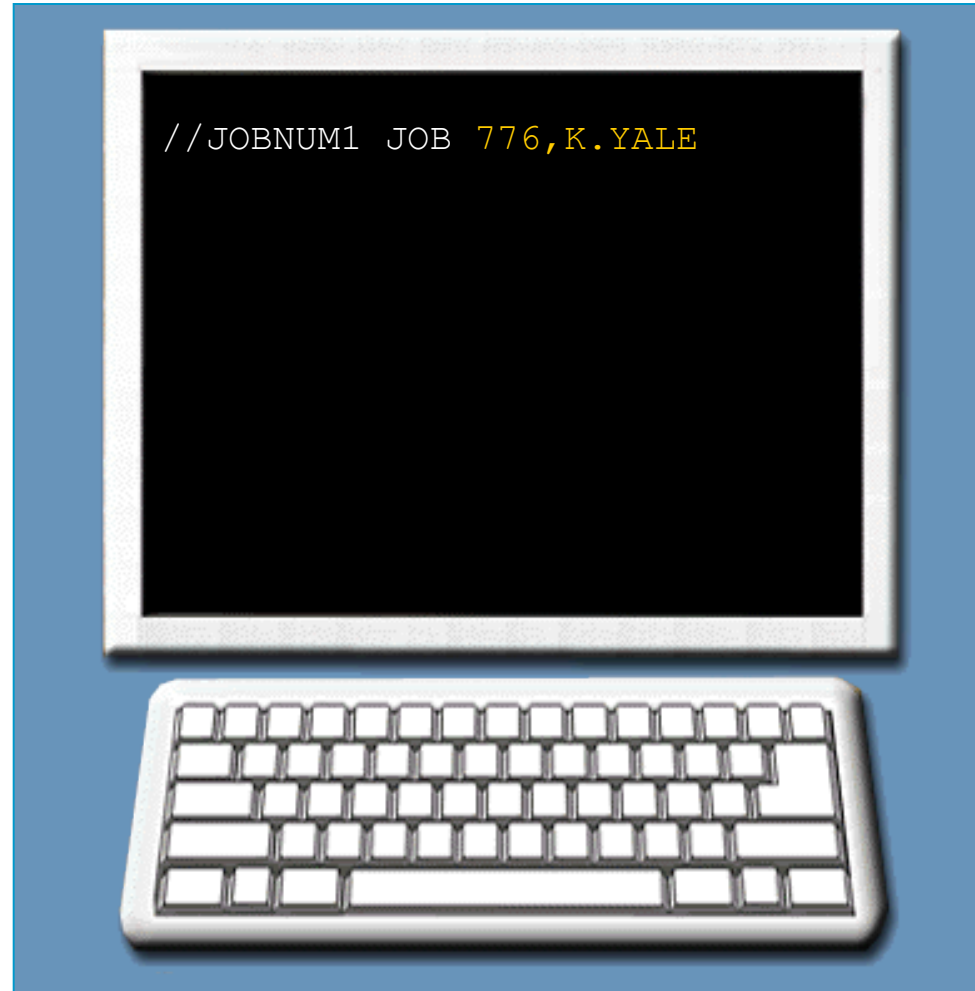
Coding the positional parameter.

The two positional parameters for a JOB statement are:

- Job accounting information.
- Programmer name.

Always code these parameters in the order shown here and separate them by a comma.

For example: The job statement shown here uses a job accounting number of 776 and identifies the programmer of this job as K.YALE.



The JOB parameter field.

Job accounting information.

The value that you code for job accounting information depends on your installation. The code is usually a number to identify a department or person to whom processor time is billed.

What is an installation?

It refers to a particular computing system, including the work it does and the people who manage and operate it.

In addition to the basic job accounting number, your installation may require information such as:

- **Date.**
- **Project director.**
- **Project number.**

The JOB parameter field.

Job accounting information: subparameters.

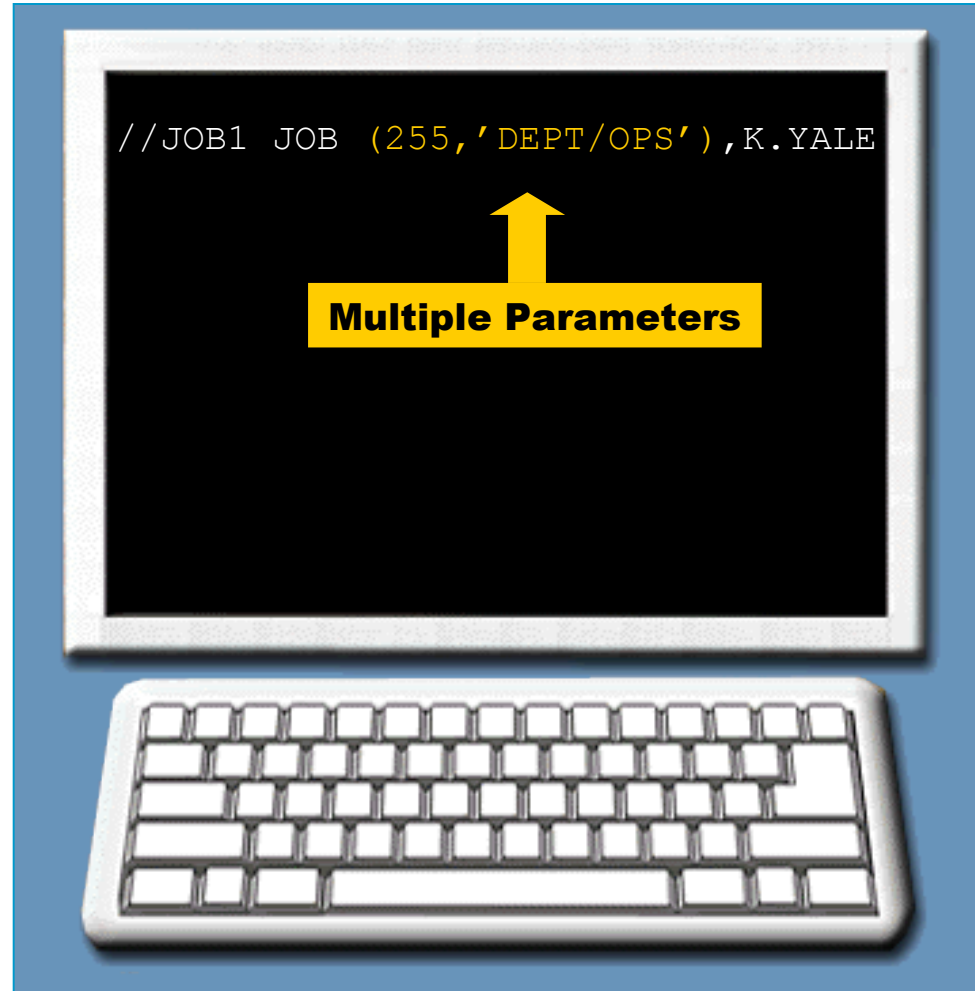
When job accounting information consists of multiple subparameters, the job accounting subparameters must be enclosed in either parenthesis or apostrophes.

The job accounting information appear in parentheses here.

Why?

Because it consists of two subparameters:

- 255
- DEPT/OPS



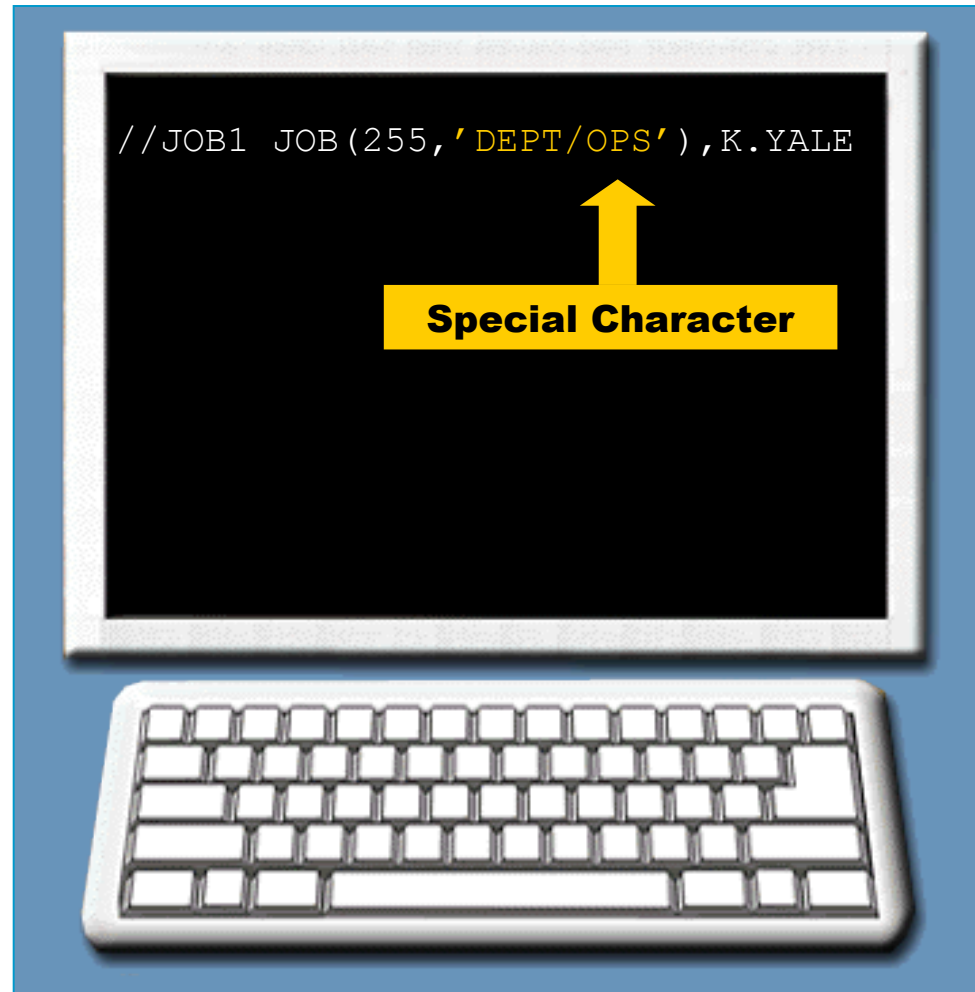
The JOB parameter field.

Job accounting information: subparameters.

Can you use special characters in subparameters?

Special characters can be used in subparameters, provided the subparameters are enclosed in apostrophes.

For example: The subparameter 'DEPT/OPS' shown is enclosed in apostrophes because the slash (/) is a special character.



The JOB parameter field.

Are we on track?

Which of the following JOB statements contain valid subparameters for job accounting information?

- A. //JOBPAY JOB 567 ,LEVEL/2,J.SMITH
- B. //JOB1 JOB (56998,'ACC/SCI'),D.LAWRENCE
- C. //UPDATE JOB '888,HQDT',JOHNSTON
- D. //INFO7 JOB (253"ZONE4"),MEADOWS

The JOB parameter field.

Are we on track?

Code multiple subparameters with an account number of 255 combined with a department identifier of DEPT/OPS

//JOB1 JOB _____

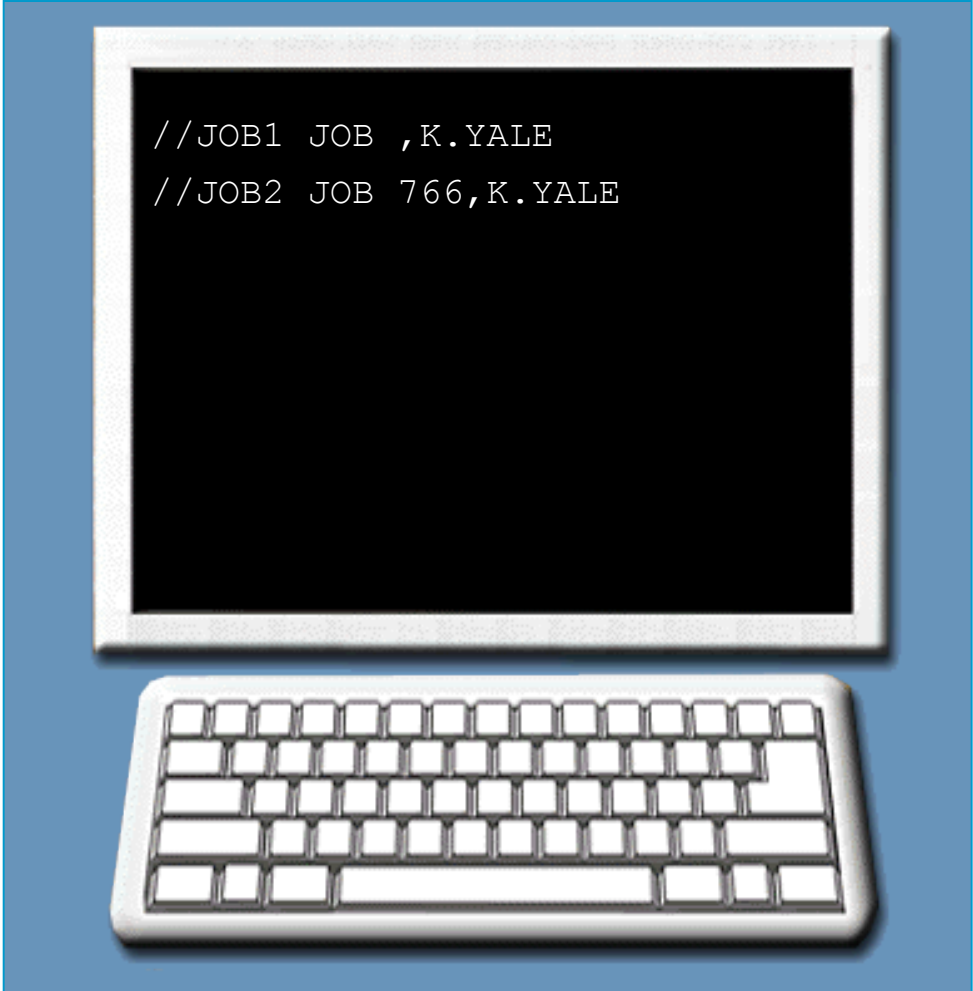
The JOB parameter field.

Omitting job accounting parameter.

Job accounting information is the first positional parameter you can code on a JOB statement.

If you omit the job accounting parameter, you must indicate its absence with a comma if you are coding the programmer name.

If you decide to leave out both positional parameters, you do not have to include commas.



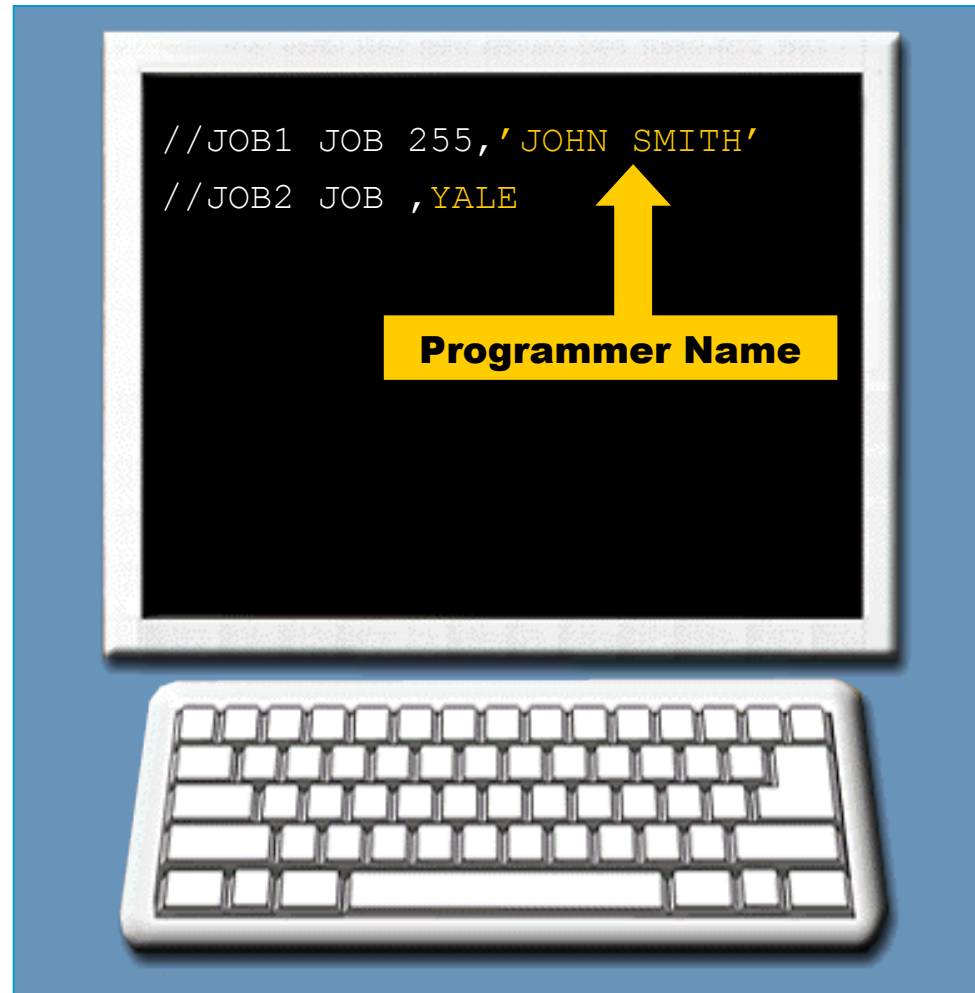
```
//JOB1 JOB ,K.YALE  
//JOB2 JOB 766,K.YALE
```

The JOB parameter field.

The programmer name.

The programmer name parameter identifies the person or group responsible for a job.

If you decide to include this information, the programmer name must immediately follow the job accounting information parameter (or a comma to indicate its absence).



The JOB parameter field.

Coding programmer name.

Programmer name coding rules:

- Separate the programmer's name from a preceding or following parameter by a comma.
- Make sure the programmer's name does not exceed 20 characters.
- Enclose the programmer's name in apostrophes when, the name contains special characters (other than periods or hyphens).
- Double any apostrophes in the programmer's name.

Programmer Name Rules

```
255, SMITH, MSGLEVEL=(1, 0)
```

```
255, LONGPROGRAMERNAME
```

```
255, 'JOHN SMITH'
```

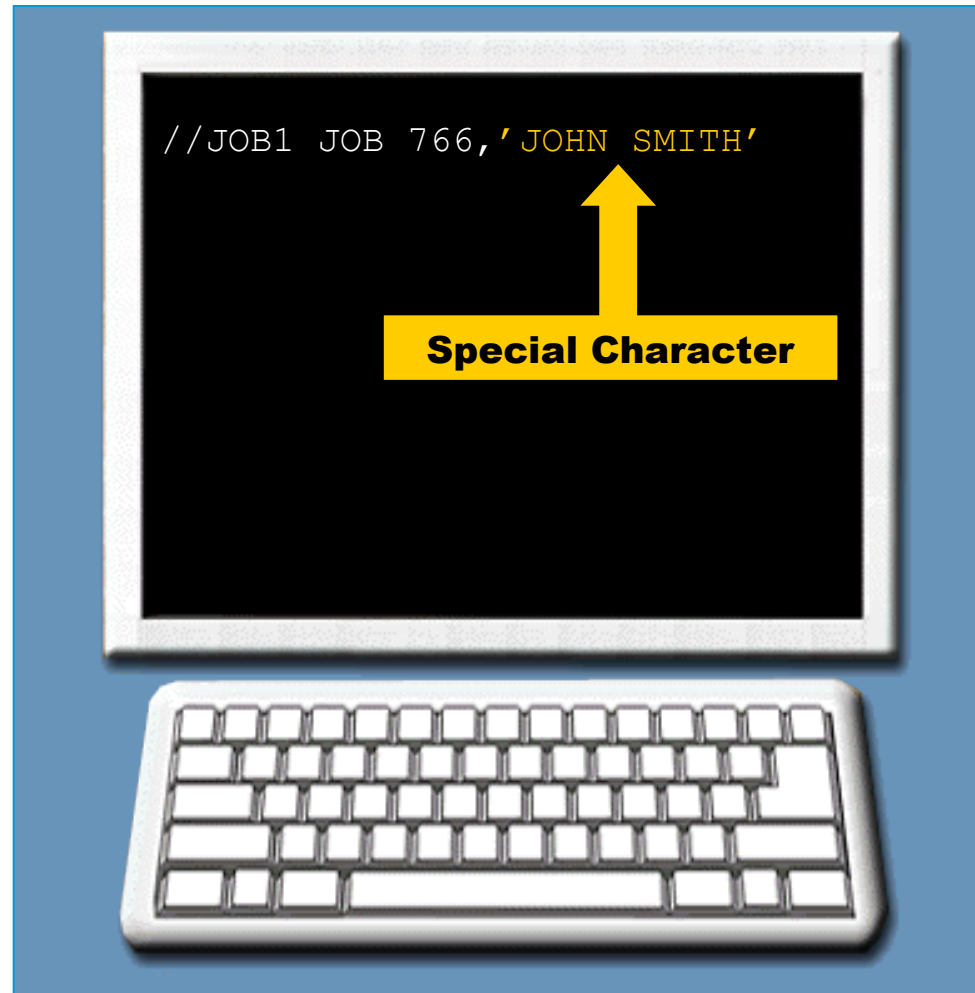
```
255, 'O''HARA'
```


The JOB parameter field.

Coding programmer name – example 1.

If you want to code the programmer name JOHN SMITH in a JOB statement, you would do it as shown here.

The name is enclosed in apostrophes because the name JOHN SMITH contains a space between the first name and the last name (a space is considered as a special character).



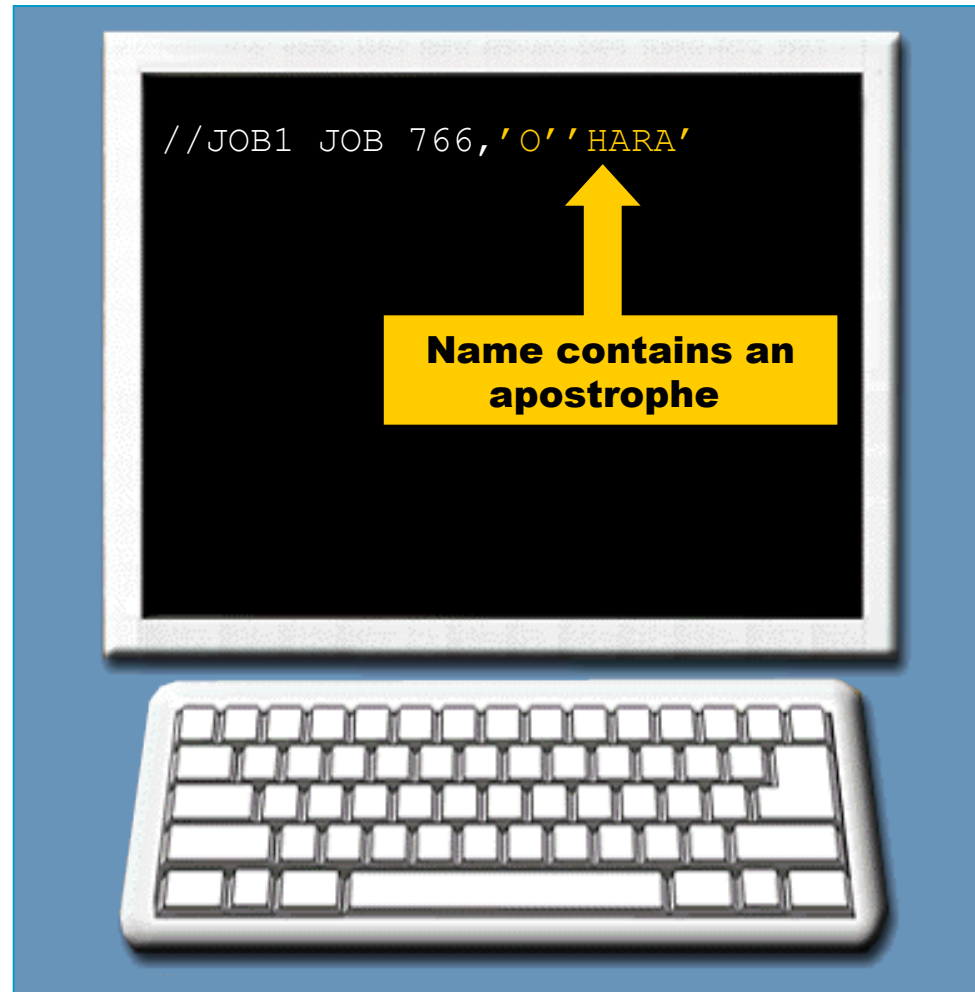
The JOB parameter field.

Coding programmer name – example 2.

If the programmer name already contains an apostrophe, you must still use apostrophes, since the apostrophe is a special character.

If you want to code the programmer name O'HARA in a JOB statement, you would do it as shown here.

If you prefer to enclose the programmer name in apostrophes, you may do so even if the name does not contain any special characters.



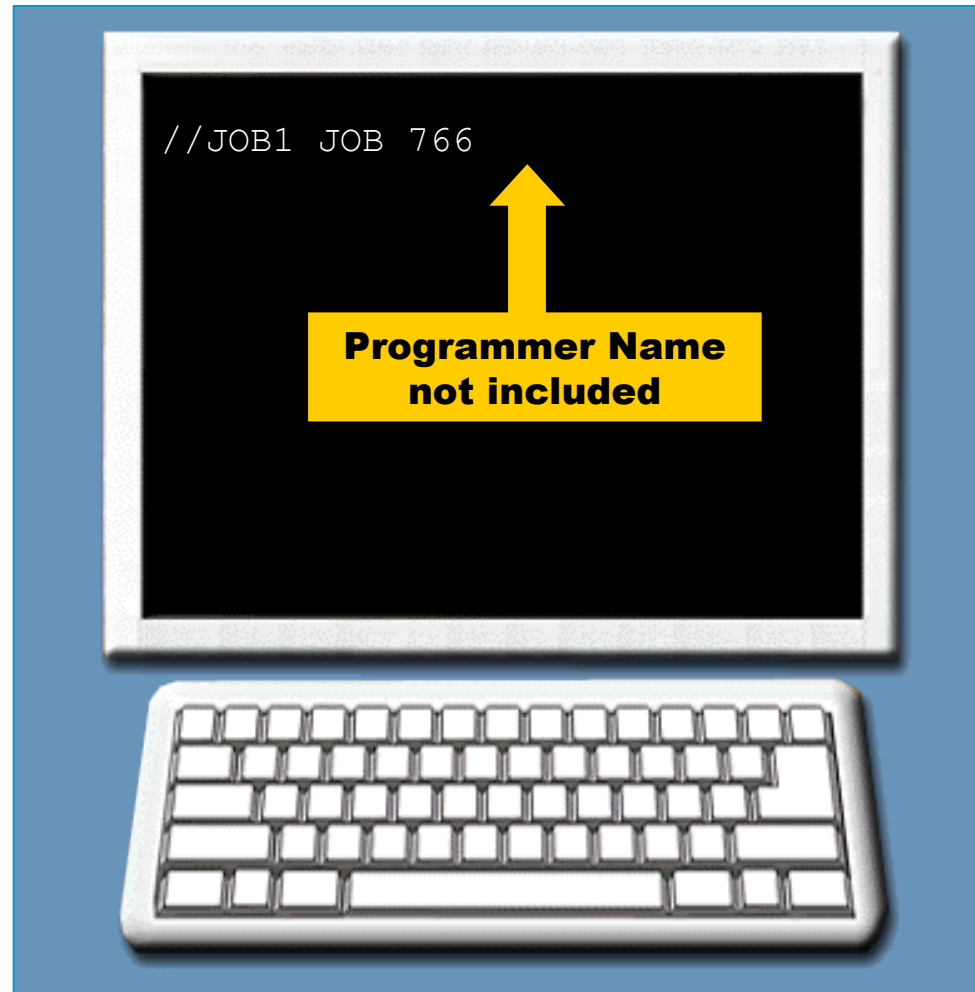
The JOB parameter field.

Ommiting programmer name.

The programmer's name is not a mandatory part of the JOB statement unless your installation has made it so.

Since the programmer name is the last positional parameter, you do not have to indicate its absence with a comma if you leave it out.

For example: The JOB statement shown here has job accounting information, but no programmer name.



The JOB parameter field.

Are we on track?

Complete the JOB statement specifying John Smith as the programmer's name.

//JOB1 JOB 255,_____

The JOB parameter field.

Are we on track?

Complete the JOB statement specifying O'Hearn as the programmer's name.

```
//JOB1 JOB 255,_____
```

The JOB parameter field.

Are we on track?

Which of the following are valid JOB statements?

A. //PAYROLLCHECKS 990,ACCOUNTING,N.JOLLIET

B. //PAY1 JOB (4456,'AP/AR')

C. //CHECK JOB ,'K.JACKSON'

D. JOB1 JOB 1298

The JOB parameter field.

Are we on track?

Which of the following rules do not apply to coding a programmer's name?

- A. Always separate the name from other parameters with a comma.**
- B. Use two consecutive apostrophes to represent an apostrophe in a name.**
- C. Use double quotation marks to enclose names with special characters.**
- D. Use as many characters as necessary in the name.**

The JOB parameter field.

Glossary.

Positional Parameters

Parameters characterized by their location in the parameter field in relation to other parameters.

Keyword Parameters

Parameters consisting of a keyword and equal sign and variable information. They do not have to be coded in a particular order.

Installation

A particular computing system, including the work it does and the people who manage and operate it.

Keyword parameters.

Defining keyword parameters.

Apart from positional parameters, the parameter may also contain keyword parameters.

What are keyword parameters?

Keyword parameters are parameters consisting of a keyword and equal sign and variable information.

The commonly used keyword parameters are:

- MSGLEVEL
- MSGCLASS

Keyword Parameters

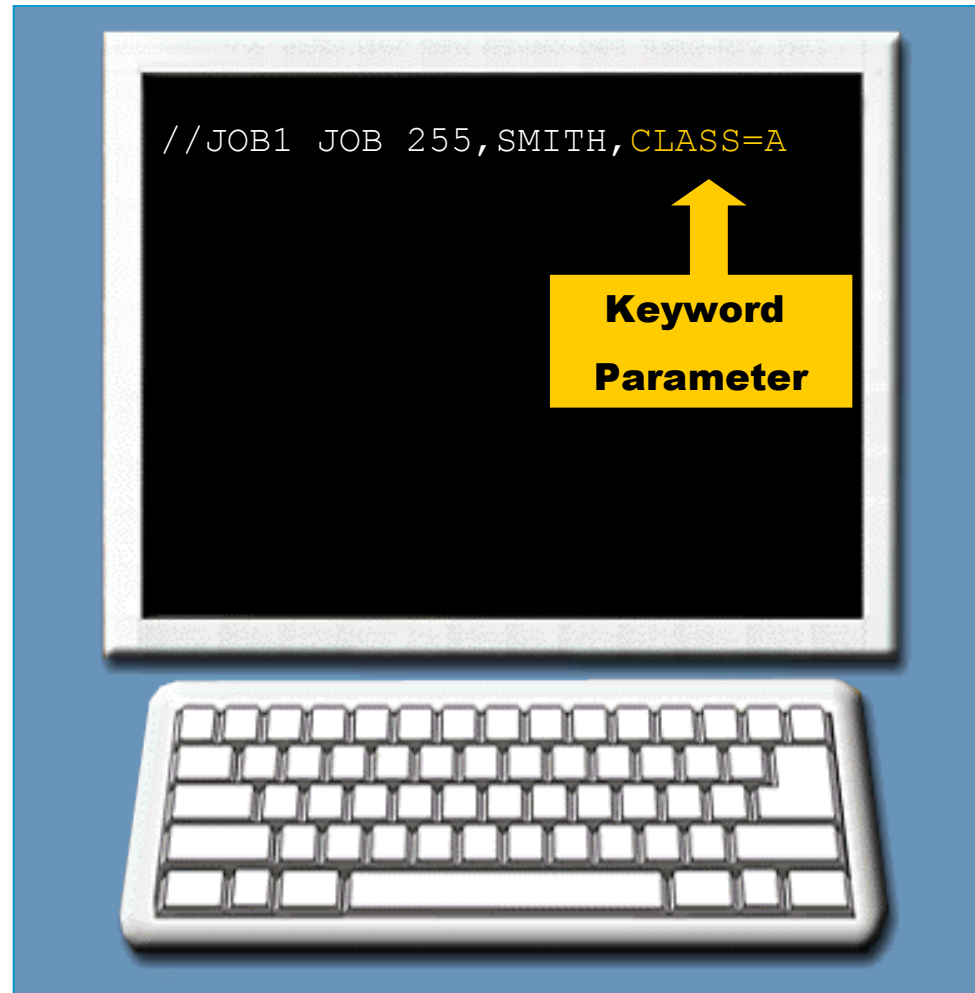
ADDRSPC	REGION
CLASS	RD
COND	LINES
GROUP	RESTART
MSGCLASS	SECLABEL
MSGLEVEL	TIME
CARDS	TYPRUN
NOTIFY	USER
PASSWORD	BYTES
PERFORM	PAGES
PRTY	SCHENV
CCSID	

Keyword parameters.

Defining keyword parameters.

The characteristics of keyword parameters include:

- They must follow any positional parameter.
- They can be coded in any order.
- They must include a keyword, an equal sign (=), and a value (for example, CLASS=A).



Keyword parameters.

The MSGLEVEL parameter.

The MSGLEVEL parameter controls how the JCL, allocation messages, and termination messages are printed in the job's output listing (SYSOUT).

You can request the following outputs using the MSGLEVEL parameter:

- **A listing of the JOB statement only.**
- **A listing of all user-supplied job control statements.**
- **A listing of all user-supplied job control statements plus all inserted statements for procedures invoked by any of the job steps.**
- **Allocation, disposition, and termination messages.**

Keyword parameters.

Coding the MSGLEVEL parameter.

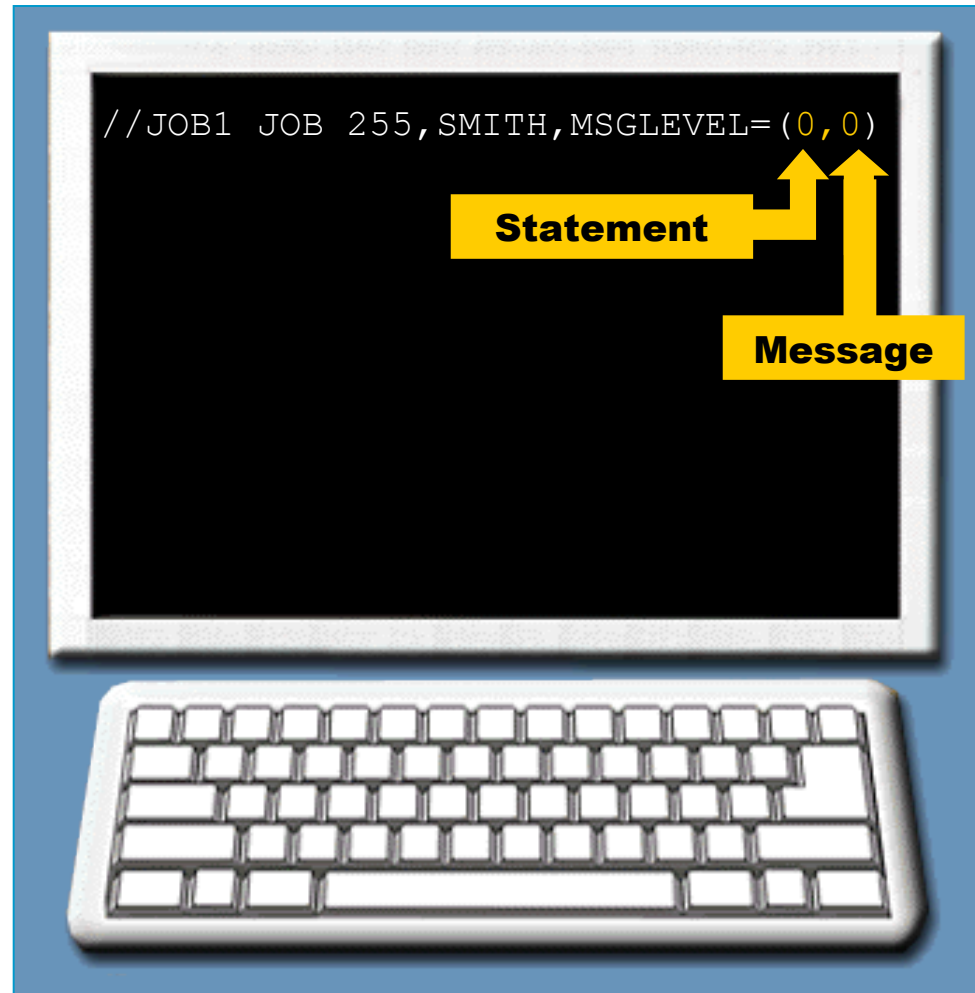
The MSGLEVEL parameter includes two subparameters:

- Statements
- Messages

The syntax for coding the MSGLEVEL parameter is:

MSGLEVEL=(statements,messages)

Multiple subparameters are enclosed in parentheses as has been shown here.



Keyword parameters.

The MSGLEVEL parameter – statement subparameter.

What does the statement subparameter indicate?

The statement subparameter indicates which job control statements the system is to print on the job log.

A statement subparameter can have one of the three values:

- **0 – Print only the JOB statement.**
- **1 – Print all JCL statements and JES2 or JES3 control statements, including invoked procedure statements.**
- **2 – Print only JCL statements and JES2 and JES3 control statements from the job stream.**

Keyword parameters.

The MSGLEVEL parameter – messages subparameter.

What does the messages subparameter indicate?

The messages subparameter indicates which messages the system is to print on the job log.

A messages subparameter can have one of the two values:

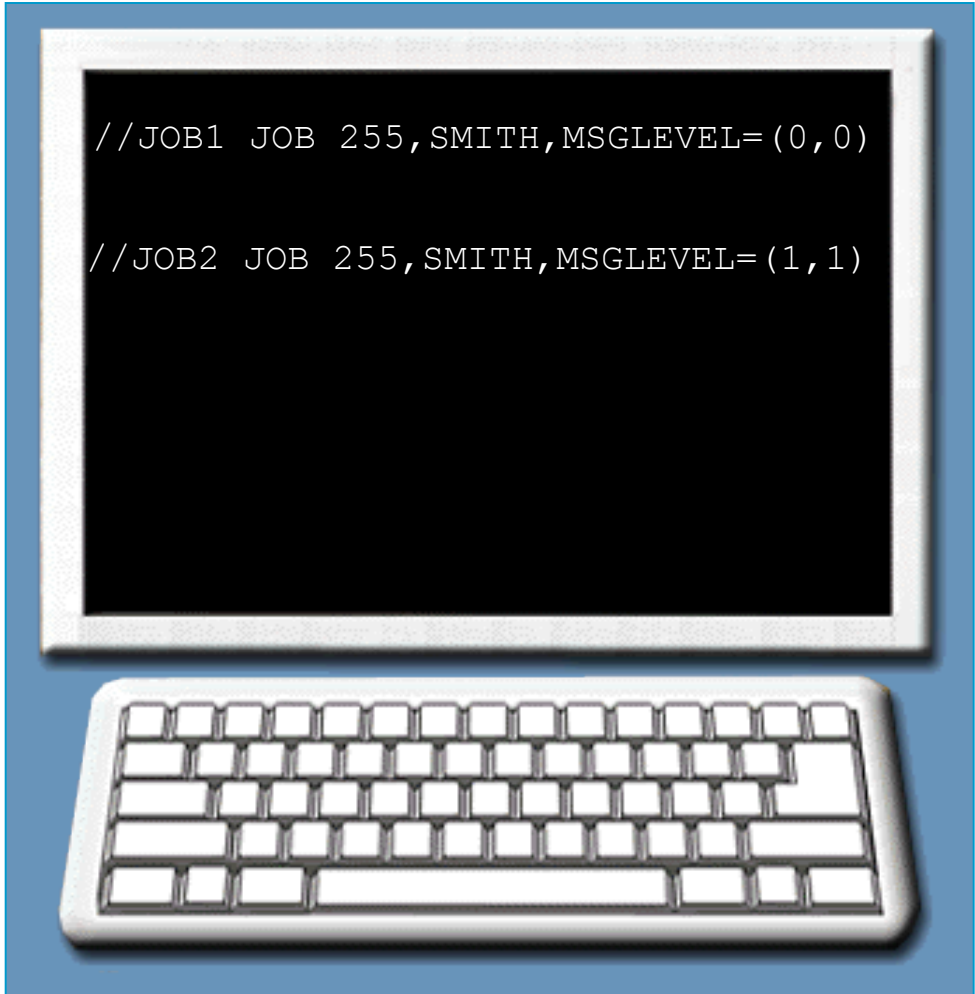
- **0 – Print only JCL messages. Print JES and operator messages only if the job terminates abnormally.**
- **1 – Print all allocation/termination messages.**

Keyword parameters.

MSGLEVEL parameter – an example.

The first JOB statement shown here specifies that only the JOB statement is to be printed and no allocation/termination messages are to be displayed with normal job execution.

In order to have the maximum display of all JCL and allocation/termination messages, the JCL coder has to code the MSGLEVEL parameter shown in the second JOB statement.



```
//JOB1 JOB 255,SMITH,MSGLEVEL=(0,0)
//JOB2 JOB 255,SMITH,MSGLEVEL=(1,1)
```

Keyword parameters.

Are we on track?

Code a MSGLEVEL parameter that prints only the JOB statement and prints all allocation/termination messages.

//JOB1 JOB 255,SMITH,_____

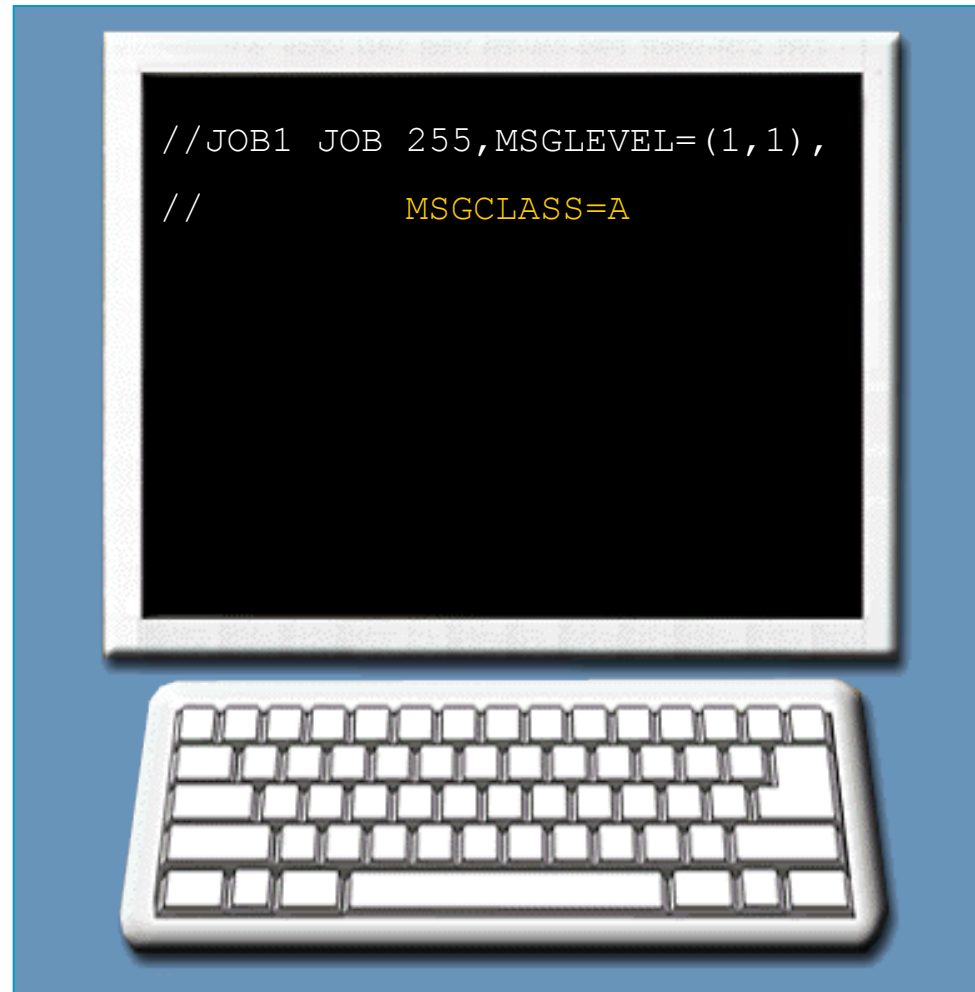
Keyword parameters.

The MSGCLASS parameter.

You can use the MSGCLASS parameter to assign an output class for your output listing (SYSOUT). Output classes are defined by the installation to designate unit record devices, such as printers.

Each class is one character long and is designated by:

- A letter (A-Z) or
- A numeral (0-9)

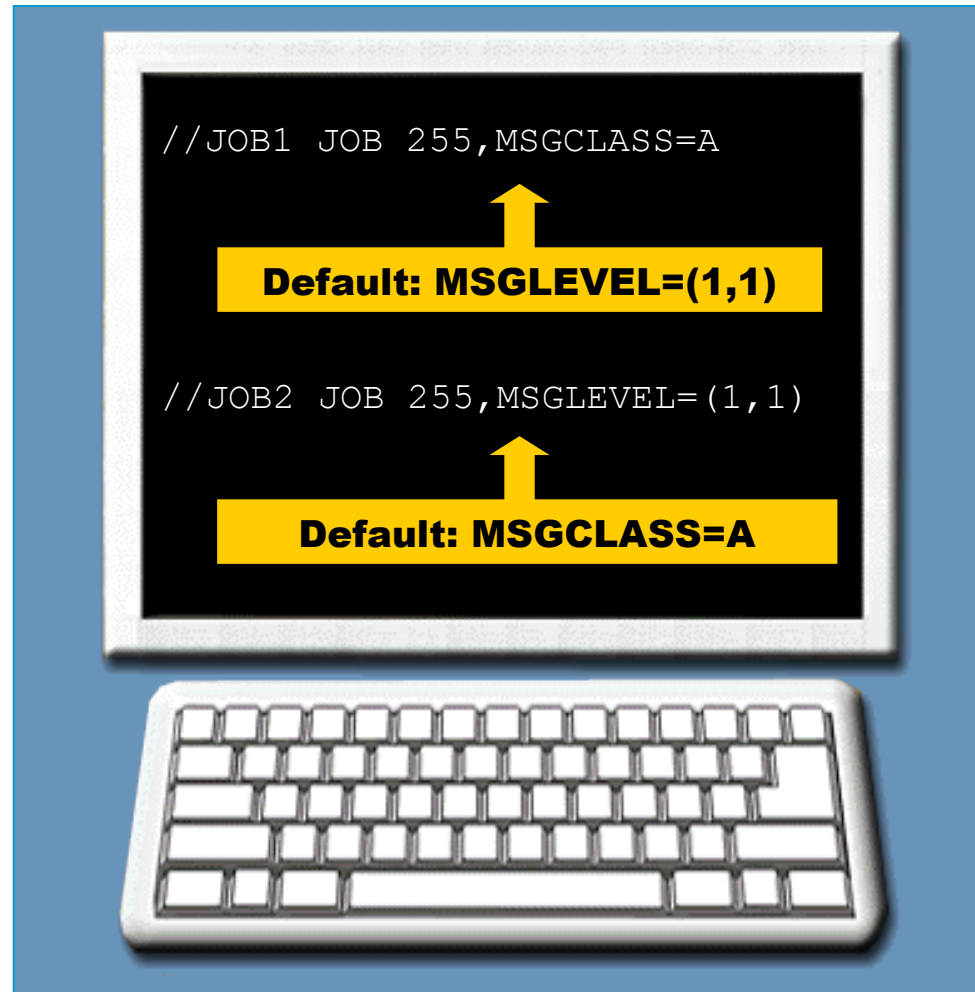


Keyword parameters.

Defaults for MSGLEVEL & MSGCLASS.

Both the MSGLEVEL and MSGCLASS parameters may have default settings, depending on your installation. Omitting one or both of the keyword parameters from the JOB statement, would make the operating system use these default settings.

In this case, you would code these parameters only if you want to have a different message level or message class than the preset.



Complete JOB statement.

JOB statement example.

In this JOB statement, the job name is JOB1, the accounting information is 776, and the programmer's name is M.FLURY.

MSGCLASS=K indicates that all the messages for this job should print on a K class output device.

MSGLEVEL=(2,1) sets the statements subparameter to 2 and the messages subparameter to 1.

A computer monitor with a white frame is shown against a blue background. The screen is black and displays two lines of white text: '//JOB1 JOB 776,M.FLURY,MSGCLASS=K,' and '// MSGLEVEL=(2,1)'. Below the monitor is a white keyboard with black keys.

```
//JOB1 JOB 776,M.FLURY,MSGCLASS=K,  
// MSGLEVEL=(2,1)
```

Complete JOB statement.

Are we on track?

Match the statement subparameter values with their descriptions.

- 1. 0 A. Print all JCL statements, control statements and invoked procedure statements.**
- 2. 1 B. Print the JOB statement only.**
- 3. 2 C. Print only JCL statements and control statements.**

Complete JOB statement.

Are we on track?

Code a MSGCLASS parameter that uses an output class of A.

//JOB1 JOB 255,SMITH,_____

Complete JOB statement.

Are we on track?

Which of the following are subparameters of the MSGLEVEL parameter?

A. Name

B. Statements

C. Class

D. Messages

Complete JOB statement.

Are we on track?

Which of the following JOB statements would O'Malley use to code JCL for the CHECKS job on account 8990 from the PAY department and to print all messages on an M class printer?

- A. //CHECKS JOB (PAY,8990)OMALLEY,MSGCLASS=(1,1),MSGLEVEL=(1,2)**
- B. //CHECKS JOB 8990,PAY,0"MALLEY,MSGLEVEL=(1,1),MSGCLASS=M**
- C. //CHECKS JOB (8990,PAY),'O'MALLEY',MSGLEVEL=(1,1),MSGCLASS=M**
- D. //CHECKS JOB (8990,PAY) OMALLEY**

JOB statement parameters.

Keyword parameters on JOB statement.

There are a host of other keyword parameters which you can use in your JOB statement.

You can use them to produce the exact results you want.

Some of the most commonly used keyword parameters have been shown here. These parameters follow the same guidelines as the keyword parameters MSGLEVEL and MSGCLASS.

```
//JOB1 JOB keyword
```



COND

ADDRSPC

CLASS

NOTIFY

PRTY

REGION

TIME

TYPRUN

USER

PASSWORD

JOB statement parameters.

The COND parameter.

The condition (COND) parameter specifies the conditions under which a job terminates.

How does the COND parameter check for a condition?

When a program terminates, it generates a return code that indicates the conditions under which the program terminated.

The system compares the value that you supply using the COND parameter with the return code of the program. This comparison determines whether or not the remaining steps in the job will be executed.

The COND parameter that you code provides the "test" needed for the comparison by supplying a value with which to compare the return code.

JOB statement parameters.

COND subparameters.

Each test requires you to code a COND parameter.

Each COND parameter uses the following two subparameters:

- **Code**
- **Operator**

Code - This subparameter specifies a decimal value to be compared with the return code provided upon completion of a program. The decimal value can range from 0 to 4095.

Operator - This subparameter specifies how the code subparameter is compared with the return code and specifies the type of text.

JOB statement parameters.

Testing of Return code.

```
//JOBNAME JOB ...,COND=(code,operator)
```

```
//JOBNAME JOB ...,COND=((code,operator),(code,operator),... )
```

You must remember to enclose each test in its own set of parentheses and the whole group of tests in another pair of parentheses, in order to test more than one return code.

You can include a maximum of eight different return code tests on each JOB statement. **If any of the tests is true, the system bypasses all the remaining steps.**

JOB statement parameters.

Are we on track?

What is the maximum number of return code tests you can code in a JOB statement condition parameter?

- A. 12**
- B. 4**
- C. 8**
- D. Unlimited**

JOB statement parameters.

The operator subparameter.

A JCL programmer can specify many ways of how the COND parameter tests a return code. The operator subparameter specifies the comparison method.

Each test in a COND subparameter specifies its own operator subparameter which is independent of operators in any other tests.

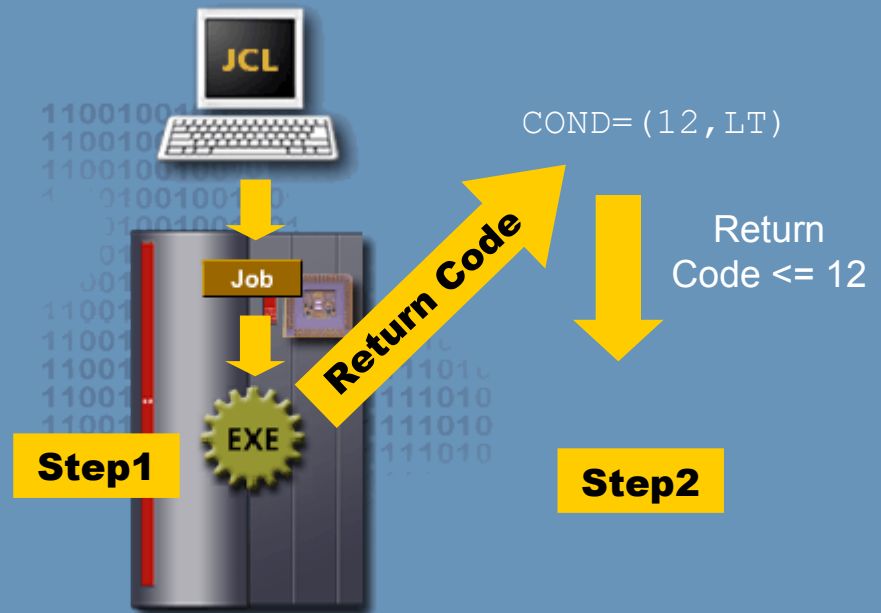
Shown here are the various operators available.

Operator	Meaning
GT	Greater than
GE	Greater than or equal to
EQ	Equal to
NE	Not equal to
LT	Less than
LE	Less than or equal to

JOB statement parameters.

Reading the COND parameter: example 1.

```
//JOB1 JOB 776, SMITH, COND=(12,LT)
//STEP1 EXEC PGM=PROGRAMA
//STEP2 EXEC PGM=PROGRAMB
```



This example helps you read the COND parameter in the JOB statement as:

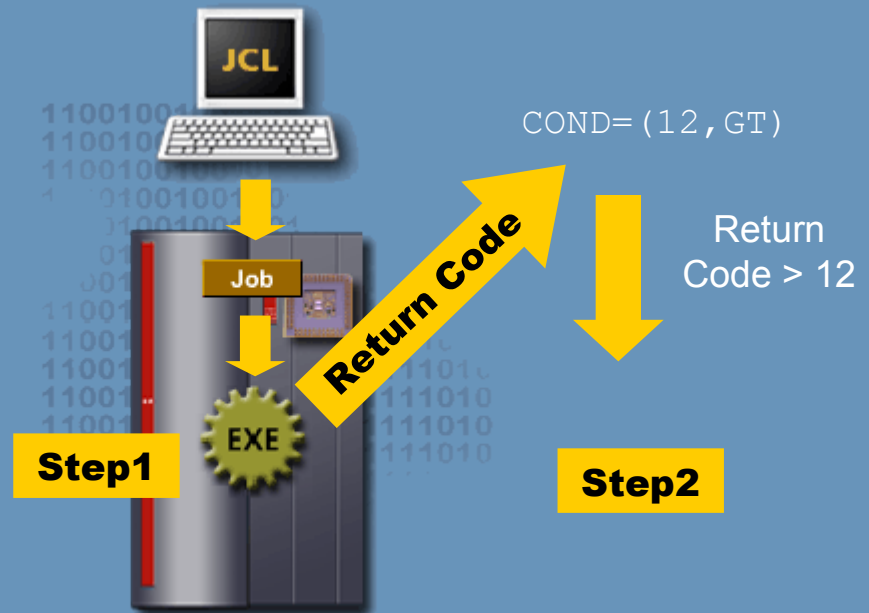
"If 12 is less than the return code, do not execute any more job steps.,,"

The job executes the remaining job steps only if the RC is 0 through 12. If the RC is 13 or higher, the remaining steps will be bypassed.

JOB statement parameters.

Reading the COND parameter: example 2.

```
//JOB2 JOB 776, SMITH, COND=(12,GT)
//STEP1 EXEC PGM=PROGRAMA
//STEP2 EXEC PGM=PROGRAMB
```



This example helps you read the COND parameter in the JOB statement as:

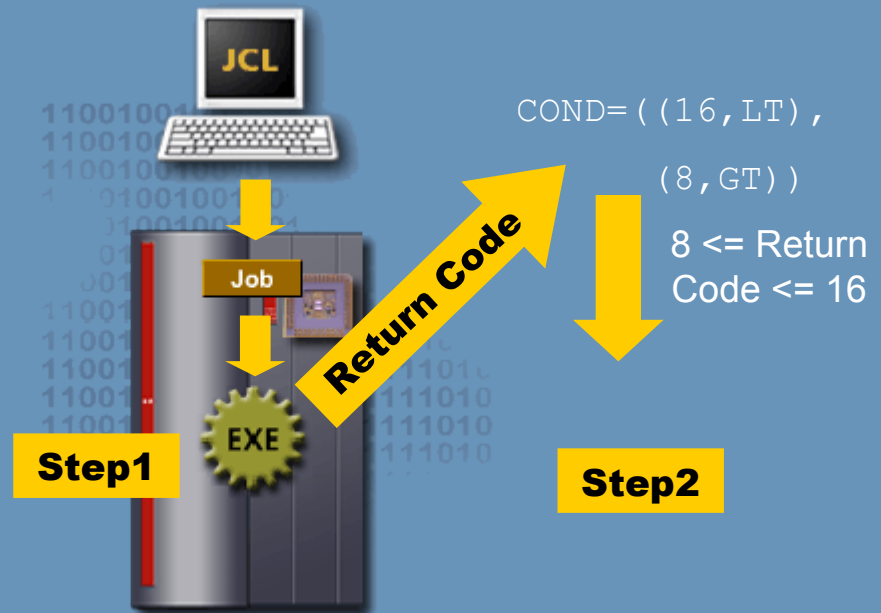
"If 12 is greater than the return code, do not execute any more job steps."

The job executes the remaining job steps only if the RC is 12 or greater. If the RC is 0 through 11, the remaining steps will be bypassed.

JOB statement parameters.

Reading the COND parameter: example 3.

```
//JOB3 JOB 776,SMITH,  
//      COND= ( (16,LT) , (8,GT) )  
//STEP1 EXEC PGM=PROGRAMA  
//STEP2 EXEC PGM=PROGRAMB
```

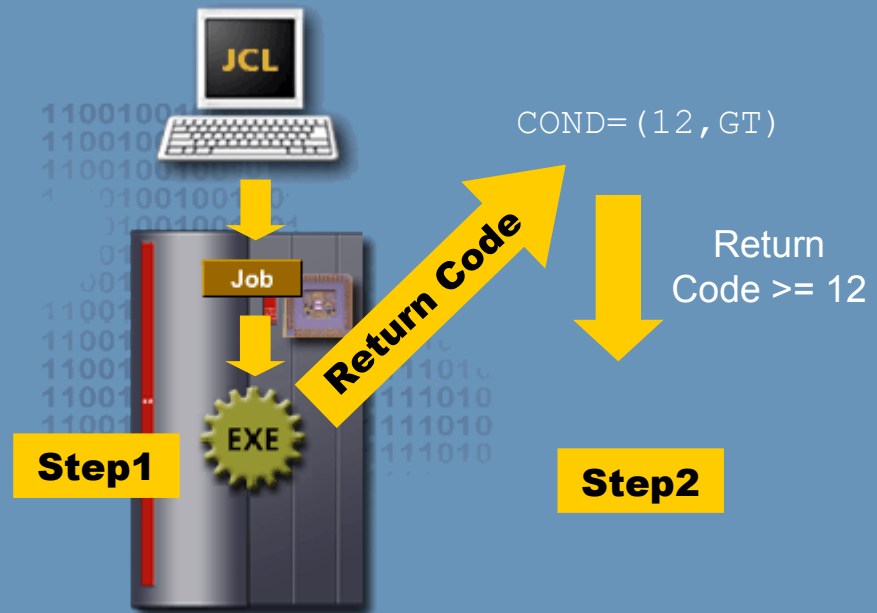


This example makes multiple comparisons. Here you can read the COND parameters in the JOB statement as: **"If 16 is less than the RC or if 8 is greater than the RC, do not execute any more job steps."** The job executes subsequent job steps only if the RC is 8-16 for each previous job step.

JOB statement parameters.

COND parameter on EXEC and JOB statements.

```
//JOB3 JOB 776,SMITH,  
// COND=( (16,LT) , (8,GT) )  
//STEP1 EXEC PGM=PROGRAMA  
//DD DD DSN=INPUT  
//STEP2 EXEC PGM=PROGRAMB,  
// COND=(12,GT)
```



You can include the COND parameter in either the EXEC statements or the JOB statements. COND parameters coded in the JOB statement are tested before any COND parameters coded in EXEC statements within the job. In the JOB statement, tests apply to all steps in the JOB.

JOB statement parameters.

Are we on track?

Code a COND parameter that will skip subsequent steps if return code of any step is greater than 12.

```
//JOB1    JOB  255,STUDENT,_____
```

JOB statement parameters.

The ADDRSPC parameter.

You use the ADDRSPC parameter when a program that the job uses should not be paged and the entire program should be placed in the nonpageable area of real storage.

Usually, the supervisor transfers the pages of a program to real storage only as needed for execution.

However, in some cases, programs need to use the nonpageable area.

JOB statement parameters.

ADDRSPC parameter values.

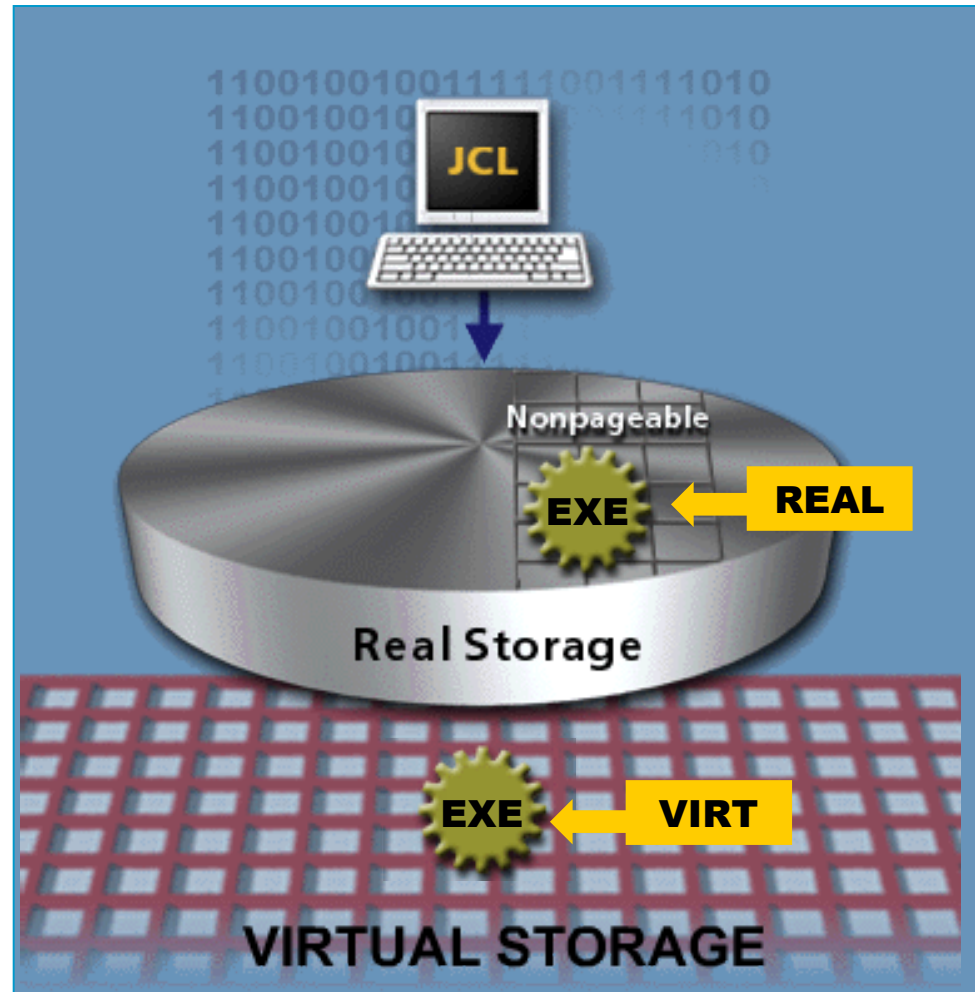
The possible values for the ADDRSPC parameters are:

```
//JOB1 JOB ,ADDRSPC={VIRT | REAL}
```

- REAL – is used to indicate that real storage must be used.
- VIRT – is used to indicate that virtual storage must be used.

If you omit the ADDRSPC parameter, the default is VIRT. Therefore, you only need to include this parameter if a job can

run only in real storage.



JOB statement parameters.

Are we on track?

Code an ADDRSPC parameter that uses REAL storage.

//JOB1 JOB 255,SMITH,_____

JOB statement parameters.

Job classes.

Why are jobs grouped into classes?

This is done for the following reasons:

- **Job classes help to achieve a balance between different types of jobs.**
- **They help avoid contention between jobs that use the same resources.**

In order to specify a particular class for your job, you have to code the CLASS parameter on your JOB statement.

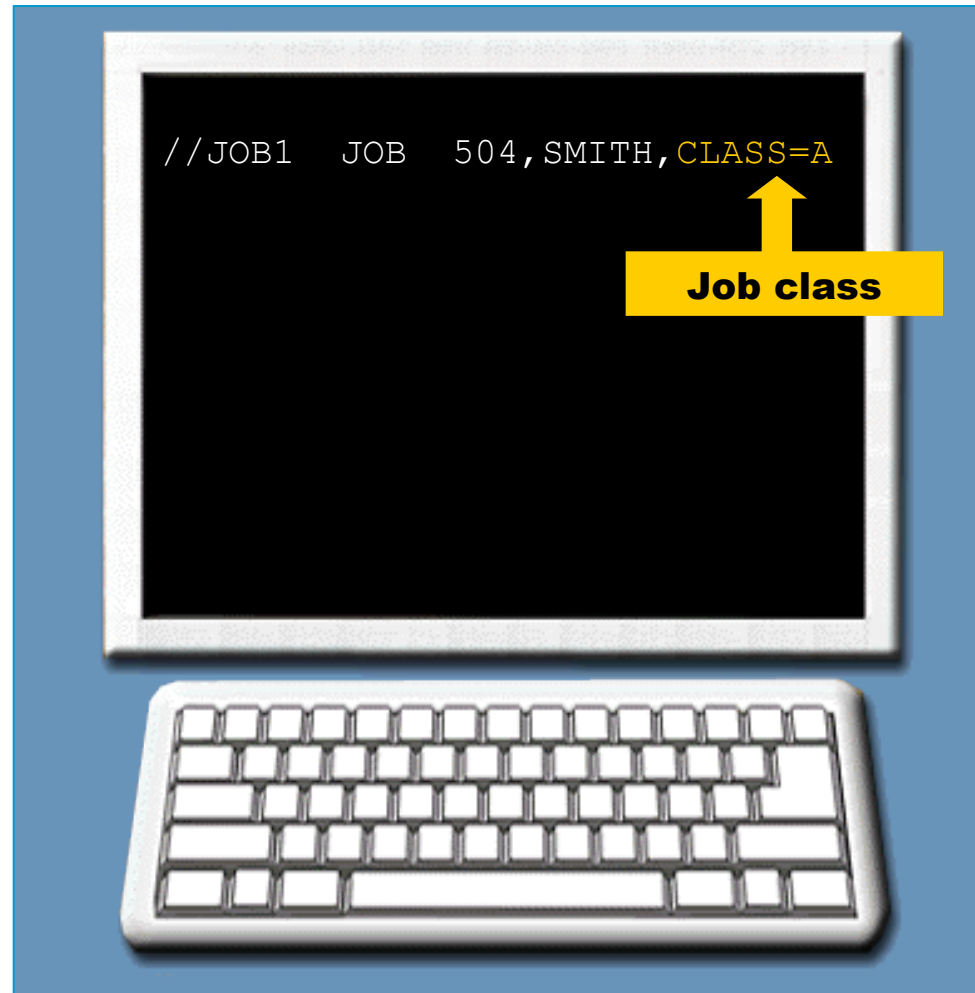
Job class is a single character subparameter that can take the values A through Z or 0 through 9.

JOB statement parameters.

The CLASS parameter.

Jobs are site-specific. You can check with your operations department about the job classes that are available for your use.

A good balance of job class assignments helps to make the most efficient use possible of the system.



JOB statement parameters.

The NOTIFY parameter.

The NOTIFY parameter indicates the TSO/E user the system must notify upon job completion. If you use the NOTIFY parameter to specify your TSO/E user ID, the operating system automatically sends you a job completion message when your job ends.

For example, to have the system send a message to JSMITH when the job EX completes, you would code the NOTIFY parameter on a JOB statement as:

```
//EX      JOB    ...,NOTIFY=JSMITH
```


JOB statement parameters.

Are we on track?

Code a NOTIFY parameter that tells the system the TSO/E user is SDJONES

//JOB1 JOB 255,SMITH,_____

JOB statement parameters.

Are we on track?

To whom does the NOTIFY parameter send a job completion message when the job ends?

A. The MVS system administrator.

B. The specified JCL programmer.

C. A specified TSO/E user.

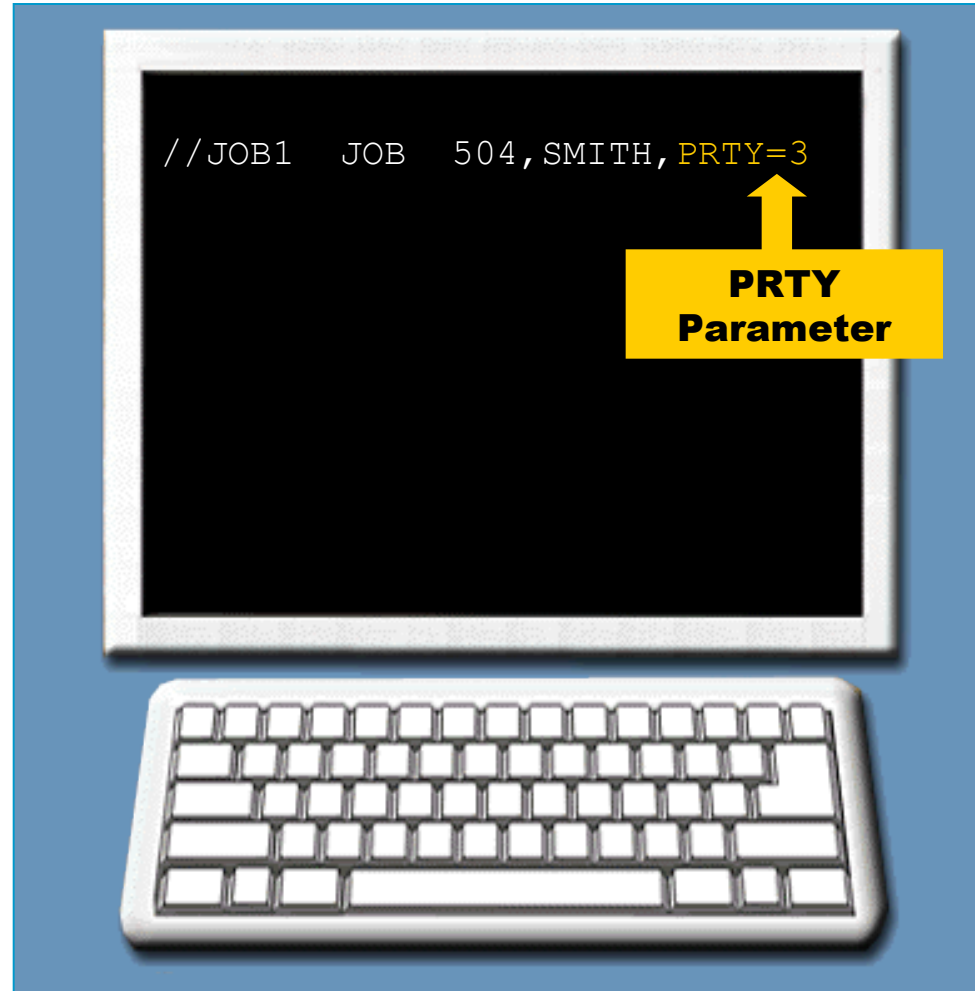
JOB statement parameters.

The PRTY parameter.

The PRTY parameter specifies a job's priority for selection within its job class. Usually, this parameter is meant to designate one job for execution over others in a class.

The range of PRTY values is usually 0 through 15, with 0 having the lowest priority.

When no priority has been specified, the system processes jobs within the same class in a first-in, first-out manner.



JOB statement parameters.

Coding CLASS and PRTY parameters.

If you have to give a specific priority to a job within a specific class, you will code both the CLASS and the PRTY parameters on the JOB statement.

For example: The JOB statement shown here specifies that the job has to run in class T and it has been accorded a priority of 3.

It is not significant in which order the CLASS and PRTY keyword parameters appear.



```
//JOB1  JOB  504, SMITH, CLASS=T,  
//                               PRTY=3
```



JOB statement parameters.

Are we on track?

Code a PRTY parameter that gives the job a specific priority of 5 (within its default class).

//JOB1 JOB 255,SMITH,_____

JOB statement parameters.

The REGION parameter.

The REGION parameter specifies the amount of storage space (in kilobytes or megabytes) that has to be allocated to a particular job.

You can make use of this parameter to override the default region size set at your installation.

Incase of a JOB statement, the region specified in the REGION parameter applies to all steps in a job and it overrides any REGION parameter coded on an EXEC statement.

```
//EXEC      JOB      ...,REGION=valueK  
//EXEC      JOB      ...,REGION=valueM
```

JOB statement parameters.

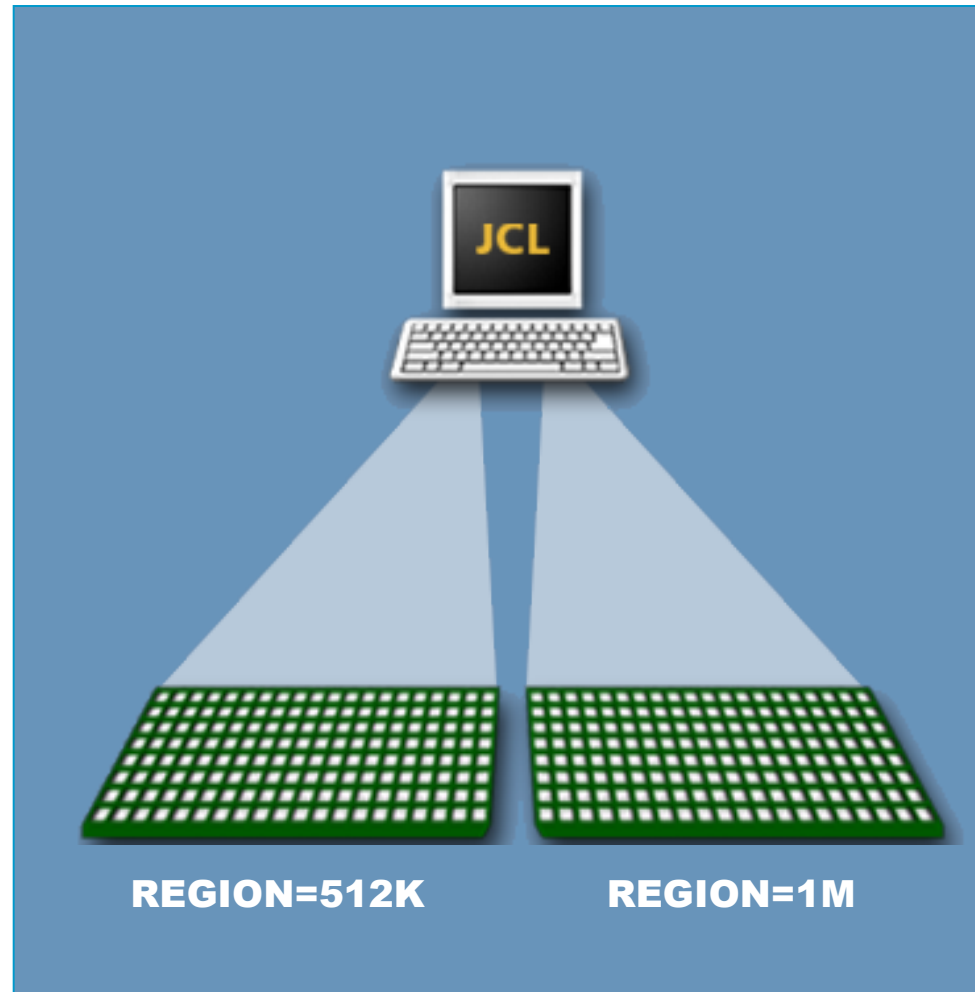
The REGION parameter.

In order to limit the virtual storage space of a job to 512 KB, you will need to code the REGION parameter on the JOB statement as:

```
//EX JOB . . . ,REGION=512K
```

In a similar manner, in order to limit a job's virtual storage space to 1024 KB bytes or 1MB, you will need to code the REGION parameter as:

```
//EX JOB . . . ,REGION=1M
```



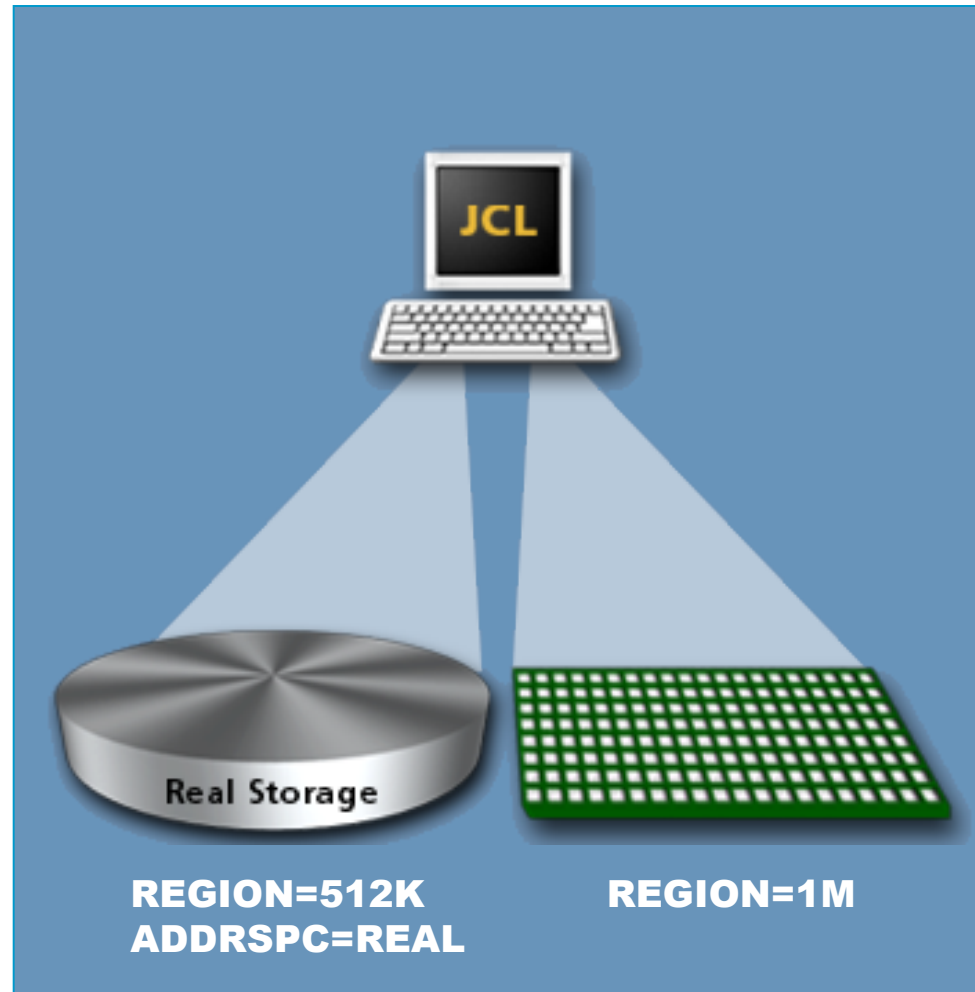
JOB statement parameters.

REGION and ADDRSPC parameters.

When you use the REGION parameter in conjunction with ADDRSPC=REAL, REGION specifies the amount of real storage.

For instance, to limit a job's real storage space to 512 KB, you code the REGION parameter, along with the ADDRSPC parameter on the JOB statement as:

```
//EX JOB . . . ,REGION=512K,  
// ADDRSPC=REAL
```



JOB statement parameters.

Are we on track?

Code a REGION parameter that limits a job's real storage space to 225K.

//JOB1 JOB 255,SMITH,_____

JOB statement parameters.

The TIME parameter.

The TIME parameter specifies a maximum amount of processor time available for the job. If the limit in the TIME parameter is reached, the job will terminate abnormally.

The syntax for the TIME parameter is:

```
//jobname JOB ...,TIME=(minutes,seconds)
```

The TIME parameter preserves processor time in case of an undetected error (like an endless loop) that may surface only during execution of program.

For example, to limit the CPU execution time to 2 minutes and 45 seconds, you will need to code the TIME parameter on a JOB statement as:

```
//EXAMPLE1 JOB 776,STUDENT,TIME=(2,45)
```

JOB statement parameters.

TIME subparameters.

Apart from the minutes and seconds subparameters, you can code three other subparameters with the TIME parameter.

These additional subparameters include

- **1440**
- **NOLIMIT**
- **MAXIMUM**

The 1440 indicates that a job is allowed to run for an unlimited amount of time. Literally, the subparameter 1440 means 1,440 minutes (i.e. 24 hours). However, the 1440 subparameter has special meaning to the operating system.

For instance, in order to allow your job to run indefinitely, you will code a JOB statement using the NOLIMIT subparameter as:

```
//EXAMPLE JOB 776,STUDENT,TIME=1440
```

JOB statement parameters.

TIME subparameters.

The NOLIMIT subparameter is identical in function to the 1440 subparameter.

Coding NOLIMIT with the TIME parameter will lead to the associated job running for an unlimited amount of time.

For instance, in order to allow a job to run indefinitely, you will have to code a JOB statement using the NOLIMIT subparameter as:

```
//EXAMPLE JOB 776,STUDENT,TIME=NOLIMIT
```

The MAXIMUM subparameter indicates that the associated job can run for 357,912 minutes, which is the maximum time the operating systems allows for a job (other than unlimited).

For example to limit the job's CPU processing time to 357,912 minutes, you would code a JOB statement as:

```
//EXAMPLE JOB 776,STUDENT,TIME=MAXIMUM
```

JOB statement parameters.

Are we on track?

Code a TIME parameter that limits CPU execution time to 1 minute and 57 seconds.

```
//JOB1    JOB 255,SMITH,_____
```

JOB statement parameters.

The TYPRUN parameter.

The TYPRUN parameter identifies jobs that have special processing requirements.

The subparameters that can be used with the TYPRUN keyword are as follows:

- **COPY (for JES2 only) - This is used to tell the system to copy the input to a SYSOUT data set for output processing, but not to execute it.**
- **HOLD - this is used to tell the system to hold the job prior to execution, until the operator releases the job.**
- **JCLHOLD (for JES2 only) - This is used to tell the system to hold the job before completing the JCL processing. JES2 holds the job until the operator releases it.**
- **SCAN - This is used to tell the tell the system to scan the JCL for syntax errors, but not to execute it.**

JOB statement parameters.

Are we on track?

Code a TYPRUN parameter that will check the JCL for syntax errors without actually executing the job.

```
//JOB1    JOB    255,SMITH,_____
//STEP1   EXEC   PGM=PROGRAMA
//DD1     DD     DSN=INPUT
```

JOB statement parameters.

The USER parameter.

The USER parameter identifies the user ID of the person who submitted the job.

This parameter uses a USERID subparameter which must be 1 to 8 alphanumeric characters or national symbols. The first character cannot be numeric.

Many system facilities, including the Resource Access Control Facility (RACF) and the System Resource Manager (SRM) use the USERID subparameter.

For example, in order to specify a user ID named HARRIS, you will have to code the USER parameter as:

```
//EXAMPLE JOB 776,STUDENT,USER=HARRIS
```

Whether the USER parameter is required by you or not, depends on the requirements of your site. In most cases, the USER parameter is used in conjunction with the PASSWORD parameter.



JOB statement parameters.

The PASSWORD parameter.

The PASSWORD parameter identifies a current RACF password for a job.

The PASSWORD parameter uses a job-specific subparameter, which can be 1 to 8 alphanumeric characters or national symbols.

Just as in the case of the USER parameter, the PASSWORD parameter may or may not be required at your site.

```
//EXAMPLE JOB ...,PASSWORD=password
```

To specify a password of XYZ123, with a user ID of HARRIS, you would code the PASSWORD and USER parameters as shown here.

```
//EXAMPLE JOB 776,STUDENT,  
//          PASSWORD=XYZ123,USER=HARRIS
```

JOB statement parameters.

Are we on track?

Complete the JOB statement by specifying a password of 123 and a user ID as JOHN.

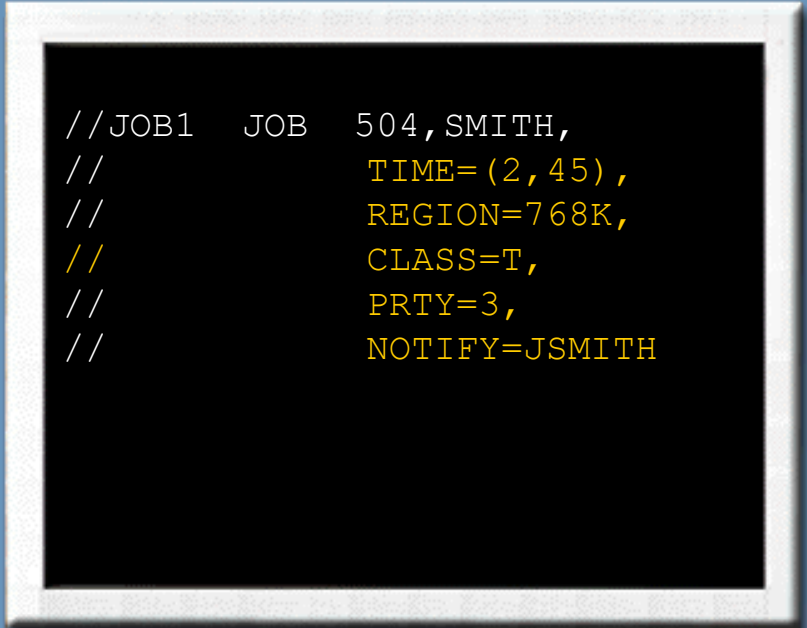
//JOB1 JOB 255,SMITH,_____

JOB statement parameters.

Coding multiple keyword parameters.

You can code multiple keyword parameters on a single JOB statement in any order that you choose after any positional parameters are coded.

Here, the JOB1 job is limited in processing time to 2 minutes, 45 seconds, and has been allocated a limited space of 768 KB. This job has a job priority of 3 within class T. This means that job named JOB1 will run ahead of all other jobs in class T that have a lower priority value (PRTY value) than 3.



```
// JOB1  JOB  504, SMITH,  
//          TIME=(2,45),  
//          REGION=768K,  
//          CLASS=T,  
//          PRTY=3,  
//          NOTIFY=JSMITH
```



JOB statement parameters.

Glossary.

Virtual Storage

Storage space that may be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses.

TSO/E

Time Sharing Option Extensions. In the MVS/ESA environment, TSO/E provides virtual storage constraint relief.

RACF

Resource Access Control Facility. An IBM licensed program that provides for access control by identifying and verifying the users of the system, by authorizing access to protected resources, by logging the detected unauthorized attempts to enter the system, and by logging the detected accesses to protected resources.

JOB statement parameters.

Unit summary.

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

- **Explain the purpose and syntax of the JOB statement.**
- **Define the JOB name.**
- **Code positional parameters and keyword parameters.**
- **Code the most commonly used keyword parameters on a JOB statement.**
- **Code additional JOB statement parameters.**

JCL

Chapter a2 Coding JOB statements

Job Control Language

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

Job Control Language

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

Job Control Language

Chapter c1. Nested procedures

Chapter c2. Cataloging procedures

Chapter c3. Using utility programs

Chapter c4. Sample utility application

Chapter a2

Coding JOB statements

A JOB statement:

- Marks the start of a particular job and gives the name of the job.
- Provides parameters and various details related to the overall job, such as accounting information and conditions for job termination.

This unit will explain how to code a simple JOB statement and include the most commonly used parameters.

Coding JOB statements

Course objectives.

Be able to:

- **Explain the purpose and syntax of the JOB statement.**
- **Define the JOB name.**
- **Code positional parameters and keyword parameters.**
- **Code the most commonly used keyword parameters on a JOB statement.**
- **Complete the JOB statement.**
- **Code additional JOB statement parameters.**

The JOB statement

Defining a JOB statement.

A JOB statement is the first statement in any JCL code.

It marks the start of a job and gives the name of the job.

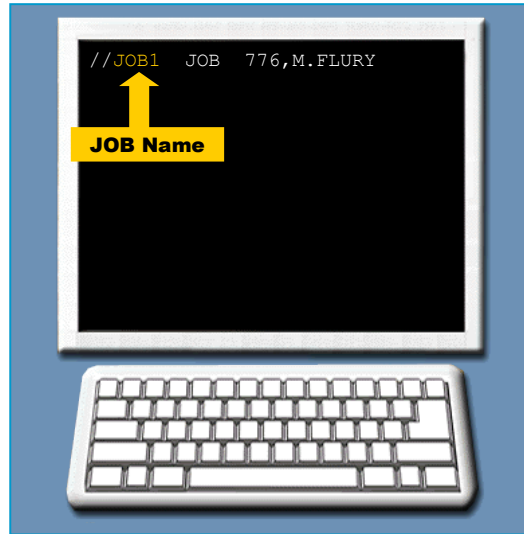
The parameters included in the JOB statement, such as accounting information and condition settings for job termination, apply to the entire job.

The JOB statement

The JOB name.

The JOB name is a 1 to 8 character name that identifies the job so that other JCL statements or the operating system can refer to it.

The JOB name must begin in position 3 with no spaces between it and the identifier.



The JOB statement.

Are we on track?

Code the JOB name as JOB1 including the identifier field and add the correct operation field.

_____ 776,M.FLURY

9

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is //JOB1 JOB

The JOB statement.

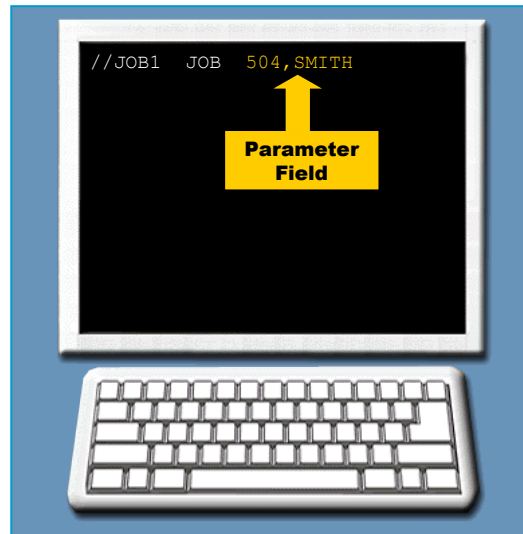
Parameter fields.

The parameter field, in the JOB statement defines information that applies to the entire job.

This information includes accounting number, programmer name, and additional information regarding the job.

One or more spaces separate the job parameters from the JOB operation field.

Always use a comma to separate parameters in a JOB statement.



10

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

Many organizations require the accounting information in order to charge computer time to the appropriate department.

Often organizations require the programmer name information so that any problems return to the appropriate individual or group.

The JOB statement.

Are we on track?

**Code the parameter field with job accounting information as 776
and the programmer's name as M. Flury.**

//JOB1 JOB _____

11

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is 776,M.FLURY

The JOB name

Nature of a job name.

The job name is the second field in a JOB statement. It follows the identifier field (/ /).

A JCL programmer should select the mandatory job name to identify the job to the operating system.

The operating system will not run jobs having the same name concurrently. Therefore, it is important that each job should be assigned a unique name.

If two jobs having the same name try to execute at the same time, the second one gets delayed till the first one completes.

See MCOE.EDU.JCL.JCL(JOBSTR).

The JOB name

Rules for job names: a summary.

Rules require a job name to begin in position 3 and to be 1 to 8 characters in length.

The first character of a job name should be either alphabetic or a national symbol. It should not be a number.

The rest of the characters in the job name can be either alphanumeric or they can be national symbols.

Special characters or spaces are not allowed.

Valid Job Names

```
//JOB1      JOB
//EXAMPLE4  JOB
//RUN#2     JOB
```

Invalid Job Names

```
//JOB1+     JOB (Includes a special character)
//EXAMPLE14 JOB (More than eight characters)
// RUN#2    JOB (Does not begin in position 3)
```

© 2013 Oracle and/or its affiliates. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The JOB name.

Are we on track?

Place the following parts of a JOB statement in the correct order.

- A. JOB**
- B. //**
- C. JOBNAME**
- D. PROGRAMMER**
- E. ACCOUNT**

14

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct order is B., C., A., E., and D.

The JOB parameter field.

Positional parameters.

The parameter field of a JOB statement appears after the JOB operator field.

There are two types of parameters:

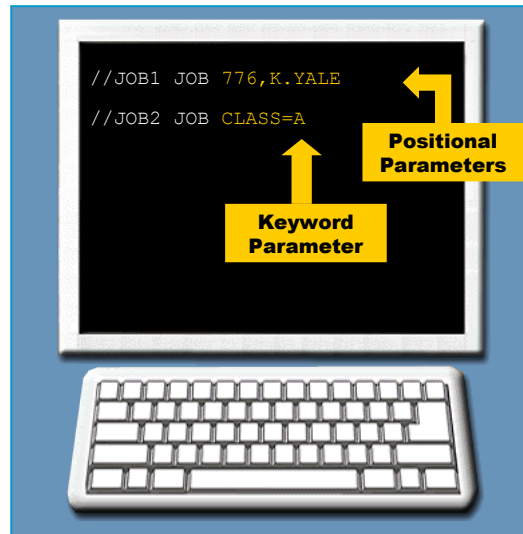
- Positional parameters.
- Keyword parameters.

What are positional parameters?

Positional parameters are characterized by their location in the parameter field in relation to other parameters.

15

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.



Positional parameters appear first in the parameter field in a fixed order if there are multiple parameters.

The JOB parameter field.

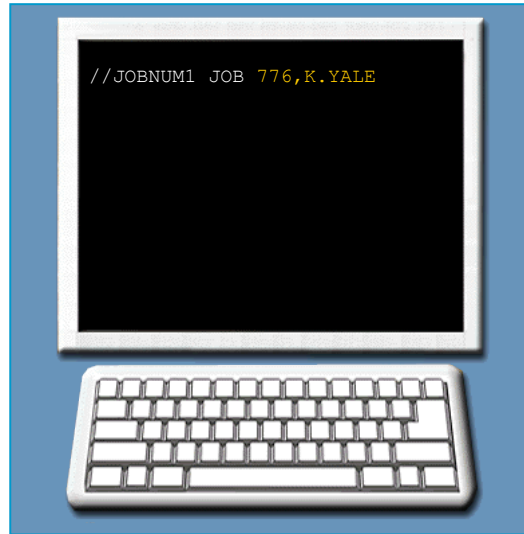
Coding the positional parameter.

The two positional parameters for a JOB statement are:

- Job accounting information.
- Programmer name.

Always code these parameters in the order shown here and separate them by a comma.

For example: The job statement shown here uses a job accounting number of 776 and identifies the programmer of this job as K.YALE.



The JOB parameter field.

Job accounting information.

The value that you code for job accounting information depends on your installation. The code is usually a number to identify a department or person to whom processor time is billed.

What is an installation?

It refers to a particular computing system, including the work it does and the people who manage and operate it.

In addition to the basic job accounting number, your installation may require information such as:

- **Date.**
- **Project director.**
- **Project number.**

The JOB parameter field.

Job accounting information: subparameters.

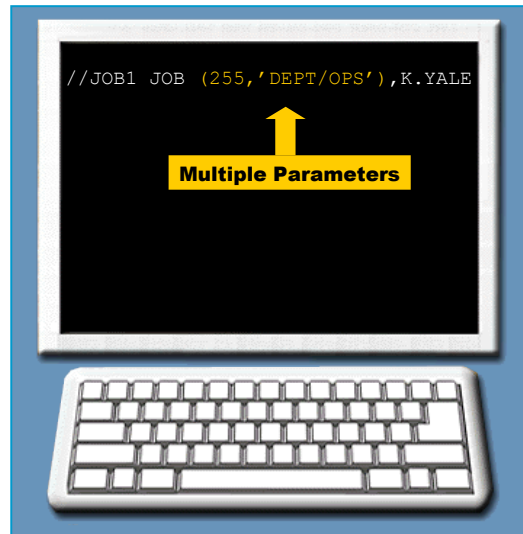
When job accounting information consists of multiple subparameters, the job accounting subparameters must be enclosed in either parenthesis or apostrophes.

The job accounting information appear in parentheses here.

Why?

Because it consists of two subparameters:

- 255
- DEPT/OPS



18

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The parentheses indicate to the operating system that both subparameters comprise the job accounting information.

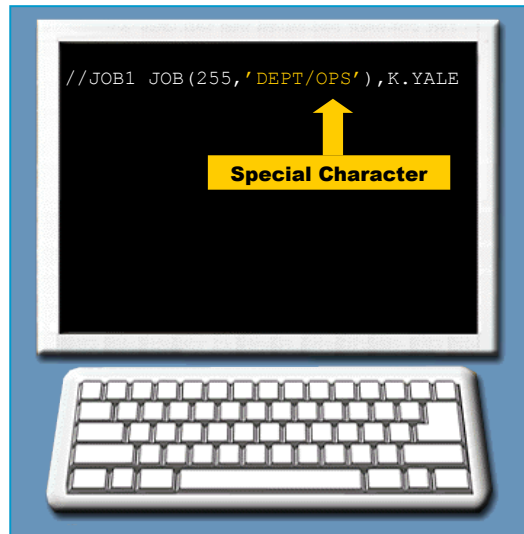
The JOB parameter field.

Job accounting information: subparameters.

Can you use special characters in subparameters?

Special characters can be used in subparameters, provided the subparameters are enclosed in apostrophes.

For example: The subparameter 'DEPT/OPS' shown is enclosed in apostrophes because the slash (/) is a special character.



If your job accounting information had an account number of 225 and a department identifier of DEPT/OPS, you would code the accounting information on the JOB statement as has been shown in here.

The JOB parameter field.

Are we on track?

Which of the following JOB statements contain valid subparameters for job accounting information?

- A. //JOBPAY JOB 567 ,LEVEL/2,J.SMITH
- B. //JOB1 JOB (56998,'ACC/SCI'),D.LAWRENCE
- C. //UPDATE JOB '888,HQDT',JOHNSTON
- D. //INFO7 JOB (253"ZONE4"),MEADOWS

20

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is B. and C.

The JOB parameter field.

Are we on track?

Code multiple subparameters with an account number of 255 combined with a department identifier of DEPT/OPS

//JOB1 JOB _____

21

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is (255,'DEPT/OPS')

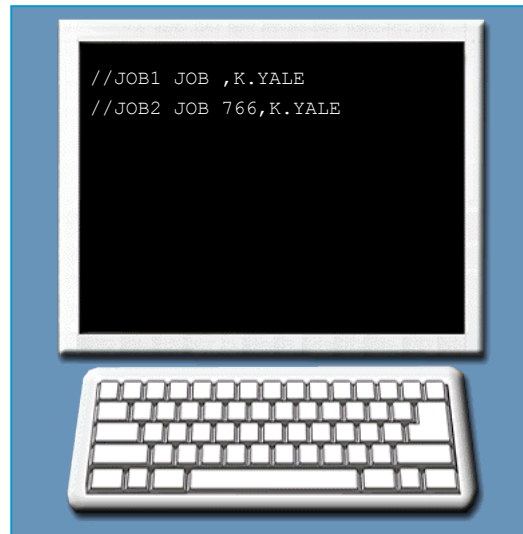
The JOB parameter field.

Omitting job accounting parameter.

Job accounting information is the first positional parameter you can code on a JOB statement.

If you omit the job accounting parameter, you must indicate its absence with a comma if you are coding the programmer name.

If you decide to leave out both positional parameters, you do not have to include commas.



22

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

For example:

In the first JOB statement here, a comma replaces the job accounting information.

The second JOB statement includes a job account number.

Both JOB statements are valid.

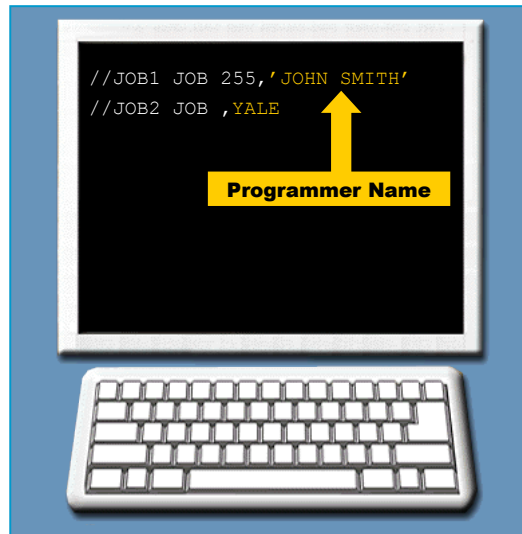
Most sites use the job accounting parameter in some form. However, this parameter is not mandatory unless your installation has modified the operating system to make the parameter mandatory.

The JOB parameter field.

The programmer name.

The programmer name parameter identifies the person or group responsible for a job.

If you decide to include this information, the programmer name must immediately follow the job accounting information parameter (or a comma to indicate its absence).



The JOB parameter field.

Coding programmer name.

Programmer name coding rules:

- Separate the programmer's name from a preceding or following parameter by a comma.
- Make sure the programmer's name does not exceed 20 characters.
- Enclose the programmer's name in apostrophes when, the name contains special characters (other than periods or hyphens).
- Double any apostrophes in the programmer's name.

Programmer Name Rules

```
255, SMITH, MSGLEVEL= (1, 0)
```

```
255, LONGPROGRAMERNAME
```

```
255, 'JOHN SMITH'
```

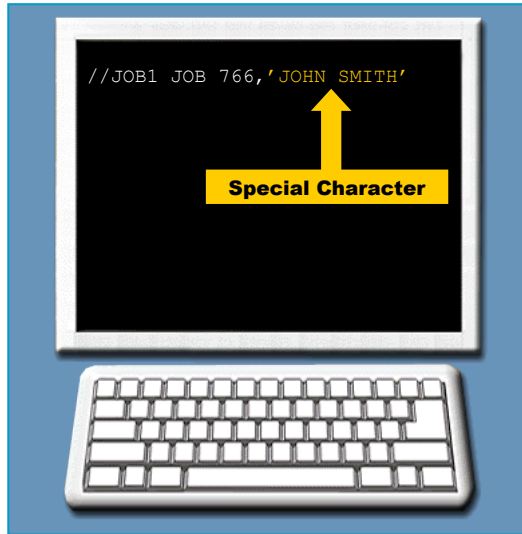
```
255, 'O' 'HARA'
```

The JOB parameter field.

Coding programmer name – example 1.

If you want to code the programmer name JOHN SMITH in a JOB statement, you would do it as shown here.

The name is enclosed in apostrophes because the name JOHN SMITH contains a space between the first name and the last name (a space is considered as a special character).



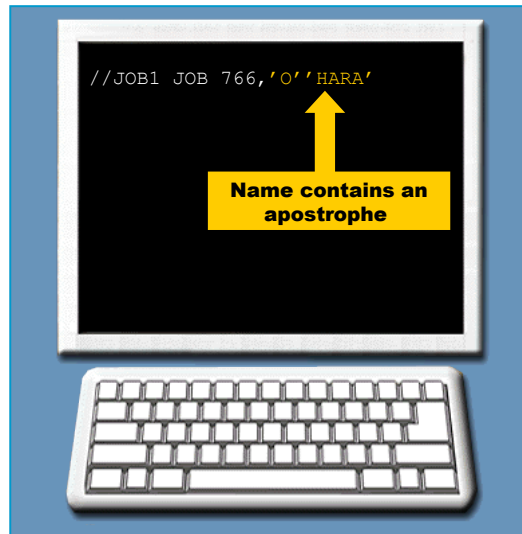
The JOB parameter field.

Coding programmer name – example 2.

If the programmer name already contains an apostrophe, you must still use apostrophes, since the apostrophe is a special character.

If you want to code the programmer name O'HARA in a JOB statement, you would do it as shown here.

If you prefer to enclose the programmer name in apostrophes, you may do so even if the name does not contain any special characters.



26

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

You must use a quotation mark to represent a single apostrophe. MVS interprets the quotation mark as an apostrophe, not as a syntactic apostrophe delimiting a string.

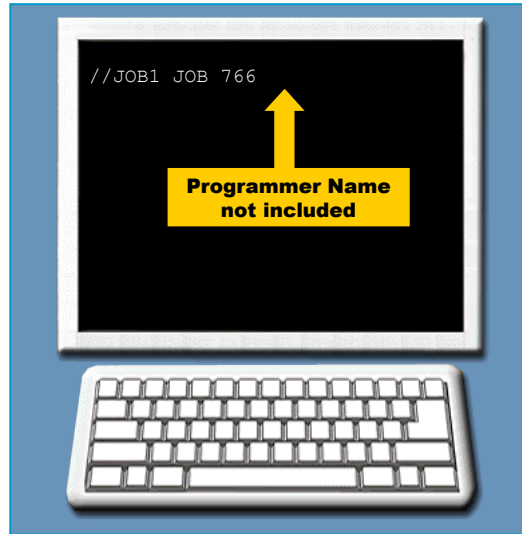
The JOB parameter field.

Ommiting programmer name.

The programmer's name is not a mandatory part of the JOB statement unless your installation has made it so.

Since the programmer name is the last positional parameter, you do not have to indicate its absence with a comma if you leave it out.

For example: The JOB statement shown here has job accounting information, but no programmer name.



27

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The JOB parameter field.

Are we on track?

Complete the JOB statement specifying John Smith as the programmer's name.

//JOB1 JOB 255,_____

28

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is 'JOHN SMITH'

The JOB parameter field.

Are we on track?

Complete the JOB statement specifying O'Hearn as the programmer's name.

//JOB1 JOB 255,_____

29

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is 'O'HEARN'

The JOB parameter field.

Are we on track?

Which of the following are valid JOB statements?

- A. //PAYROLLCHECKS 990,ACCOUNTING,N.JOLLIET
- B. //PAY1 JOB (4456,'AP/AR')
- C. //CHECK JOB ,'K.JACKSON'
- D. JOB1 JOB 1298

30

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is B. and C.

The JOB parameter field.

Are we on track?

Which of the following rules do not apply to coding a programmer's name?

- A. Always separate the name from other parameters with a comma.**
- B. Use two consecutive apostrophes to represent an apostrophe in a name.**
- C. Use double quotation marks to enclose names with special characters.**
- D. Use as many characters as necessary in the name.**

31

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is C. and D.

The JOB parameter field.

Glossary.

Positional Parameters

Parameters characterized by their location in the parameter field in relation to other parameters.

Keyword Parameters

Parameters consisting of a keyword and equal sign and variable information. They do not have to be coded in a particular order.

Installation

A particular computing system, including the work it does and the people who manage and operate it.

Keyword parameters.

Defining keyword parameters.

Apart from positional parameters, the parameter may also contain keyword parameters.

What are keyword parameters?

Keyword parameters are parameters consisting of a keyword and equal sign and variable information.

The commonly used keyword parameters are:

- MSGLEVEL
- MSGCLASS

Keyword Parameters	
ADDRSPC	REGION
CLASS	RD
COND	LINES
GROUP	RESTART
MSGCLASS	SECLABEL
MSGLEVEL	TIME
CARDS	TYPRUN
NOTIFY	USER
PASSWORD	BYTES
PERFORM	PAGES
PRTY	SCHENV
CCSID	

33

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

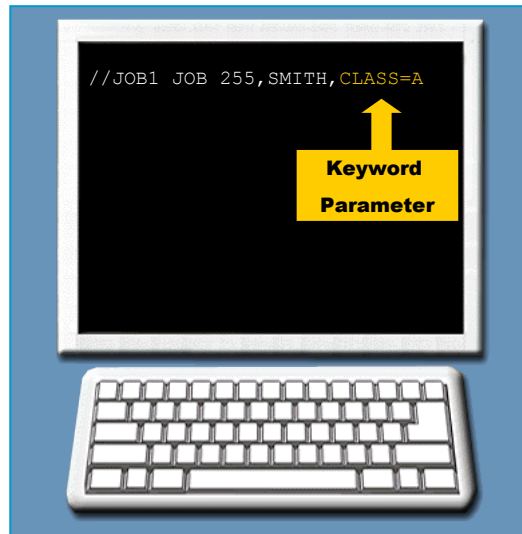
Keyword parameters supply information to the operating system for control of jobs.

Keyword parameters.

Defining keyword parameters.

The characteristics of keyword parameters include:

- They must follow any positional parameter.
- They can be coded in any order.
- They must include a keyword, an equal sign (=), and a value (for example, CLASS=A).



Keyword parameters.

The MSGLEVEL parameter.

The MSGLEVEL parameter controls how the JCL, allocation messages, and termination messages are printed in the job's output listing (SYSOUT).

You can request the following outputs using the MSGLEVEL parameter:

- **A listing of the JOB statement only.**
- **A listing of all user-supplied job control statements.**
- **A listing of all user-supplied job control statements plus all inserted statements for procedures invoked by any of the job steps.**
- **Allocation, disposition, and termination messages.**

Keyword parameters.

Coding the MSGLEVEL parameter.

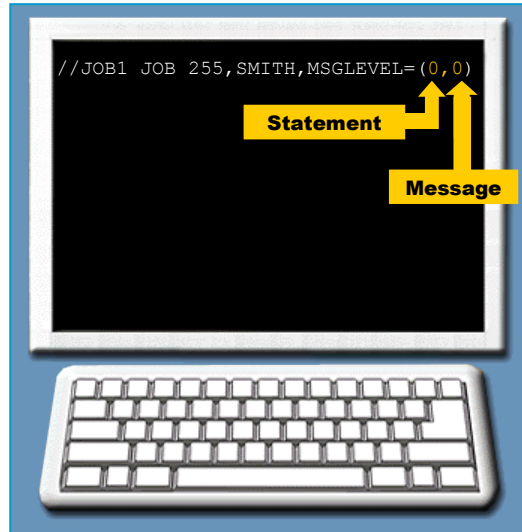
The MSGLEVEL parameter includes two subparameters:

- Statements
- Messages

The syntax for coding the MSGLEVEL parameter is:

MSGLEVEL=(statements ,messages)

Multiple subparameters are enclosed in parentheses as has been shown here.



36

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

Keyword parameters.

The MSGLEVEL parameter – statement subparameter.

What does the statement subparameter indicate?

The statement subparameter indicates which job control statements the system is to print on the job log.

A statement subparameter can have one of the three values:

- **0 – Print only the JOB statement.**
- **1 – Print all JCL statements and JES2 or JES3 control statements, including invoked procedure statements.**
- **2 – Print only JCL statements and JES2 and JES3 control statements from the job stream.**

Keyword parameters.

The MSGLEVEL parameter – messages subparameter.

What does the messages subparameter indicate?

The messages subparameter indicates which messages the system is to print on the job log.

A messages subparameter can have one of the two values:

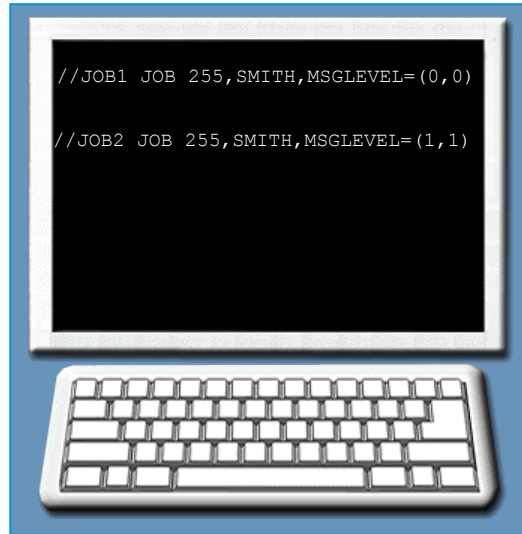
- **0 – Print only JCL messages. Print JES and operator messages only if the job terminates abnormally.**
- **1 – Print all allocation/termination messages.**

Keyword parameters.

MSGLEVEL parameter – an example.

The first JOB statement shown here specifies that only the JOB statement is to be printed and no allocation/termination messages are to be displayed with normal job execution.

In order to have the maximum display of all JCL and allocation/termination messages, the JCL coder has to code the MSGLEVEL parameter shown in the second JOB statement.



Keyword parameters.

Are we on track?

Code a MSGLEVEL parameter that prints only the JOB statement and prints all allocation/termination messages.

//JOB1 JOB 255,SMITH,_____

40

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is MSGLEVEL=(0,1)

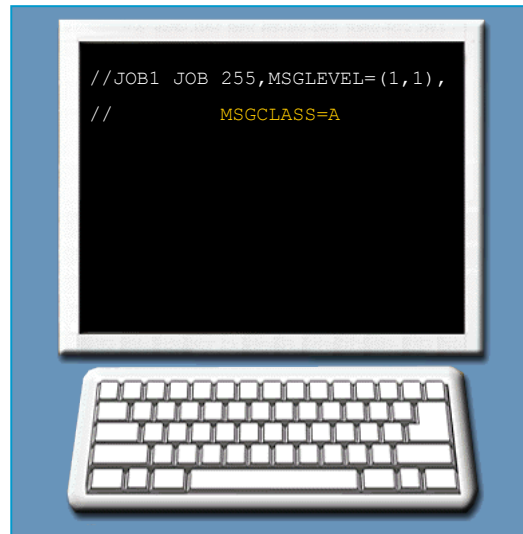
Keyword parameters.

The MSGCLASS parameter.

You can use the MSGCLASS parameter to assign an output class for your output listing (SYSOUT). Output classes are defined by the installation to designate unit record devices, such as printers.

Each class is one character long and is designated by:

- A letter (A-Z) or
- A numeral (0-9)



41

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

For example: In order to assign class A as the output class for your listing, you would code the MSGCLASS parameter as shown here.

Notice that the MSGCLASS parameter does not use parentheses because unlike the MSGLEVEL parameter, the MSGCLASS parameter has only one subparameter.

You can find the MSGCLASS definition from SYSVIEW Primary Option Menu:

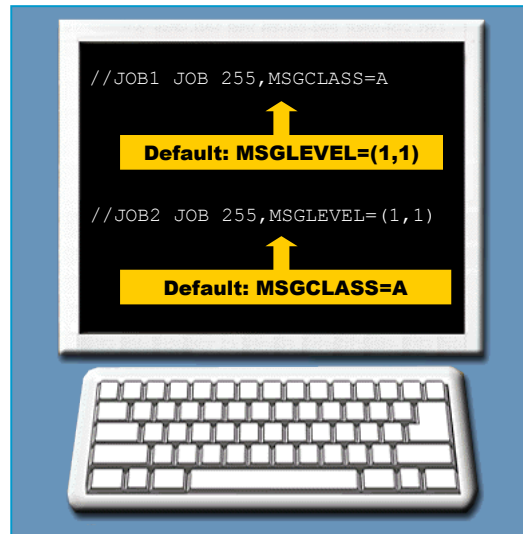
- 3 MENU JES2 JES2 job and output management
- 11 MENU JES2OUT Output management
- 16 OUTCLASS Output classes

Keyword parameters.

Defaults for MSGLEVEL & MSGCLASS.

Both the MSGLEVEL and MSGCLASS parameters may have default settings, depending on your installation. Omitting one or both of the keyword parameters from the JOB statement, would make the operating system use these default settings.

In this case, you would code these parameters only if you want to have a different message level or message class than the preset.



The MSGLEVEL subparameters are universal but output class assignments and the default settings for both parameters depend on your installation.

Complete JOB statement.

JOB statement example.

In this JOB statement, the job name is JOB1, the accounting information is 776, and the programmer's name is M.FLURY.

MSGCLASS=K indicates that all the messages for this job should print on a K class output device.

MSGLEVEL=(2,1) sets the statements subparameter to 2 and the messages subparameter to 1.



43

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

Only the JCL and the JES2 control statements from the input stream will print in the SYSOUT listing along with all allocation and termination messages.

You can code keyword parameters in any order.

Complete JOB statement.

Are we on track?

Match the statement subparameter values with their descriptions.

- | | | |
|-------------|-----------|---|
| 1. 0 | A. | Print all JCL statements, control statements and invoked procedure statements. |
| 2. 1 | B. | Print the JOB statement only. |
| 3. 2 | C. | Print only JCL statements and control statements. |

44

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is 1-B, 2-A, 3-C

Complete JOB statement.

Are we on track?

Code a MSGCLASS parameter that uses an output class of A.

```
//JOB1 JOB 255,SMITH,_____
```

45

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is MSGCLASS=A

Complete JOB statement.

Are we on track?

Which of the following are subparameters of the MSGLEVEL parameter?

A. Name

B. Statements

C. Class

D. Messages

46

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is B. and D.

Complete JOB statement.

Are we on track?

Which of the following JOB statements would O'Malley use to code JCL for the CHECKS job on account 8990 from the PAY department and to print all messages on an M class printer?

- A. //CHECKS JOB (PAY,8990)OMALLEY,MSGCLASS=(1,1),MSGLEVEL=(1,2)**
- B. //CHECKS JOB 8990,PAY,0"MALLEY,MSGLEVEL=(1,1),MSGCLASS=M**
- C. //CHECKS JOB (8990,PAY),'O"MALLEY',MSGLEVEL=(1,1),MSGCLASS=M**
- D. //CHECKS JOB (8990,PAY) OMALLEY**

47

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is C.

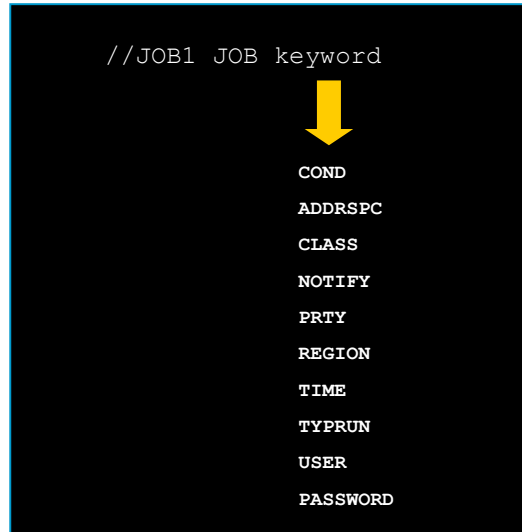
JOB statement parameters.

Keyword parameters on JOB statement.

There are a host of other keyword parameters which you can use in your JOB statement.

You can use them to produce the exact results you want.

Some of the most commonly used keyword parameters have been shown here. These parameters follow the same guidelines as the keyword parameters MSGLEVEL and MSGCLASS.



JOB statement parameters.

The COND parameter.

The condition (COND) parameter specifies the conditions under which a job terminates.

How does the COND parameter check for a condition?

When a program terminates, it generates a return code that indicates the conditions under which the program terminated.

The system compares the value that you supply using the COND parameter with the return code of the program. This comparison determines whether or not the remaining steps in the job will be executed.

The COND parameter that you code provides the "test" needed for the comparison by supplying a value with which to compare the return code.

JOB statement parameters.

COND subparameters.

Each test requires you to code a COND parameter.

Each COND parameter uses the following two subparameters:

- **Code**
- **Operator**

Code - This subparameter specifies a decimal value to be compared with the return code provided upon completion of a program. The decimal value can range from 0 to 4095.

Operator - This subparameter specifies how the code subparameter is compared with the return code and specifies the type of text.

JOB statement parameters.

Testing of Return code.

```
//JOBNAME JOB ...,COND=(code,operator)

//JOBNAME JOB ...,COND=((code,operator),(code,operator),... )
```

You must remember to enclose each test in its own set of parentheses and the whole group of tests in another pair of parentheses, in order to test more than one return code.

You can include a maximum of eight different return code tests on each JOB statement. **If any of the tests is true, the system bypasses all the remaining steps.**

Attempting to code more than eight comparisons will result in you receiving a JCL error message and termination of the job abnormally.

JOB statement parameters.

Are we on track?

What is the maximum number of return code tests you can code in a JOB statement condition parameter?

- A. 12**
- B. 4**
- C. 8**
- D. Unlimited**

52

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is C.

JOB statement parameters.

The operator subparameter.

A JCL programmer can specify many ways of how the COND parameter tests a return code. The operator subparameter specifies the comparison method.

Each test in a COND subparameter specifies its own operator subparameter which is independent of operators in any other tests.

Shown here are the various operators available.

Operator	Meaning
GT	Greater than
GE	Greater than or equal to
EQ	Equal to
NE	Not equal to
LT	Less than
LE	Less than or equal to

53

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

Overrides If you code the COND parameter on the JOB statement and on one or more of the job's EXEC statements, and if a return code test on the JOB statement is satisfied, the job terminates. In this case, the system ignores any EXEC statement COND parameters.

JOB statement parameters.

Reading the COND parameter: example 1.

```
//JOB1 JOB 776, SMITH, COND= (12, LT)
//STEP1 EXEC PGM=PROGRAMA
//STEP2 EXEC PGM=PROGRAMB
```

COND= (12, LT)
Return Code <= 12

Step1 Step2

This example helps you read the COND parameter in the JOB statement as:

"If 12 is less than the return code, do not execute any more job steps.."

The job executes the remaining job steps only if the RC is 0 through 12. If the RC is 13 or higher, the remaining steps will be bypassed.

It means:

RC is 0-11: consequently 12 is GT RC consequently execute the remaining steps.

RC is 12: consequently 12 is EQ RC consequently execute the remaining steps.

RC is 13: consequently 12 is LT RC consequently bypass the remaining steps.

Try MCOE.EDU.JCL.JCL(CONDSAMP).

If condition is met, bypass all remaining steps.

JOB statement parameters.

Reading the COND parameter: example 2.

This example helps you read the COND parameter in the JOB statement as:

"If 12 is greater than the return code, do not execute any more job steps."

The job executes the remaining job steps only if the RC is 12 or greater. If the RC is 0 through 11, the remaining steps will be bypassed.

It means:

- RC is 0-11: consequently 12 is GT RC consequently bypass the remaining steps.
- RC is 12: consequently 12 is EQ RC consequently execute the remaining steps.
- RC is 13: consequently 12 is LT RC consequently execute the remaining steps.

Try MCOE.EDU.JCL.JCL(CONDSAMP).

JOB statement parameters.

Reading the COND parameter: example 3.

```
//JOB3 JOB 776, SMITH,  
// COND= ((16,LT), (8,GT))  
//STEP1 EXEC PGM=PROGRAMA  
//STEP2 EXEC PGM=PROGRAMB
```

COND= ((16,LT), (8,GT))
8 <= Return Code <= 16

Step1 Step2

This example makes multiple comparisons. Here you can read the COND parameters in the JOB statement as: **"If 16 is less than the RC or if 8 is greater than the RC, do not execute any more job steps."** The job executes subsequent job steps only if the RC is 8-16 for each previous job step.

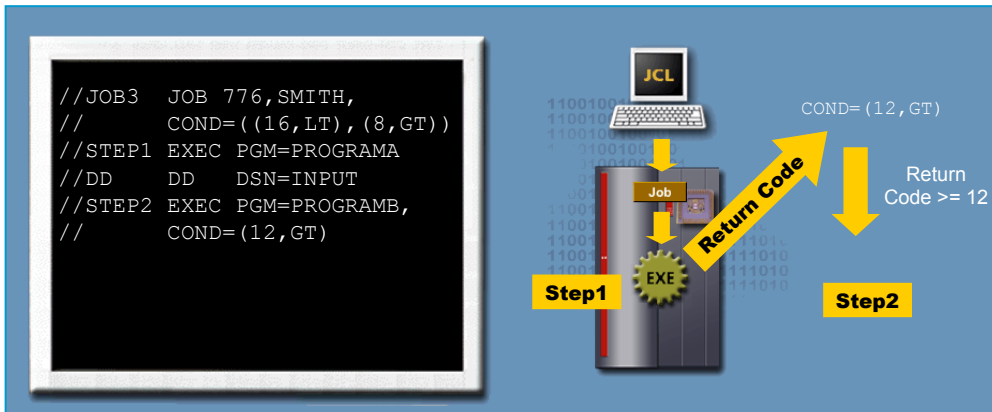
It means:

RC is 0-7: then 8 is GT RC then bypass the remaining steps.
RC is 8: then 16 is GT RC and 8 is EQ RC then execute the remaining steps.
RC is 9-15: then 16 is GT RC and 8 is LT RC then execute the remaining steps.
RC is 16: then 16 is EQ RC and 8 is LT RC then execute the remaining steps.
RC is 17: then 16 is LT RC then bypass the remaining steps.

Try MCOE.EDU.JCL.JCL(CONDSAMP).

JOB statement parameters.

COND parameter on EXEC and JOB statements.



You can include the COND parameter in either the EXEC statements or the JOB statements. COND parameters coded in the JOB statement are tested before any COND parameters coded in EXEC statements within the job. In the JOB statement, tests apply to all steps in the JOB.

See „z/OS MVS JCL Reference“ manual,

Overrides:

If you code the COND parameter on the JOB statement and on one or more of the job's EXEC statements, and if a RC test on the JOB statement is satisfied, the job terminates. In this case, the system ignores any EXEC statement COND parameters. When COND is coded in the EXEC statement, it is possible to test the RC of specific steps.

JOB statement parameters.

Are we on track?

Code a COND parameter that will skip subsequent steps if return code of any step is greater than 12.

//JOB1 JOB 255,STUDENT,_____

58

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is COND=(12,LT)

Remember: If 12 is less than the return code, do not execute any more job steps.

JOB statement parameters.

The ADDRSPC parameter.

You use the ADDRSPC parameter when a program that the job uses should not be paged and the entire program should be placed in the nonpageable area of real storage.

Usually, the supervisor transfers the pages of a program to real storage only as needed for execution.

However, in some cases, programs need to use the nonpageable area.

JOB statement parameters.

ADDRSPC parameter values.

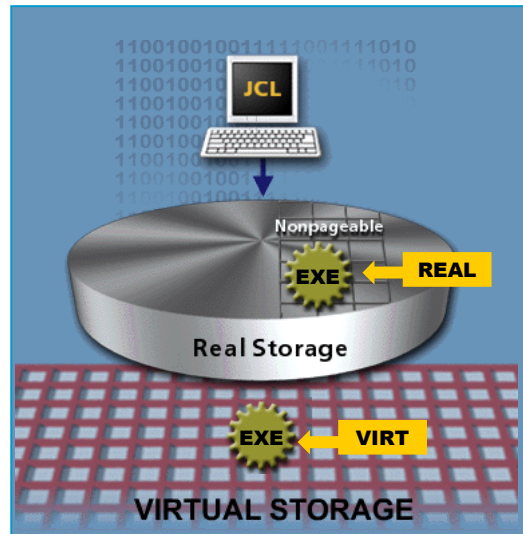
The possible values for the ADDRSPC parameters are:

```
//JOB1 JOB ,ADDRSPC={VIRT|REAL}
```

- REAL – is used to indicate that real storage must be used.
- VIRT – is used to indicate that virtual storage must be used.

If you omit the ADDRSPC parameter, the default is VIRT. Therefore, you only need to include this parameter if a job can

60 only in real storage.



Virtual storage: The storage space that may be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses.

JOB statement parameters.

Are we on track?

Code an ADDRSPC parameter that uses REAL storage.

//JOB1 JOB 255,SMITH,_____

61

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is ADDRSPC=REAL

JOB statement parameters.

Job classes.

Why are jobs grouped into classes?

This is done for the following reasons:

- **Job classes help to achieve a balance between different types of jobs.**
- **They help avoid contention between jobs that use the same resources.**

In order to specify a particular class for your job, you have to code the CLASS parameter on your JOB statement.

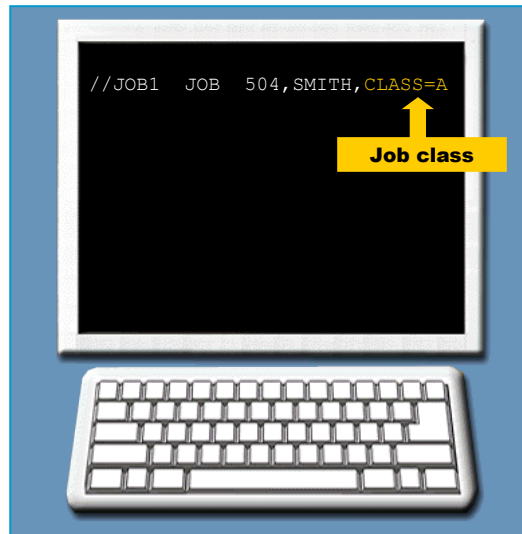
Job class is a single character subparameter that can take the values A through Z or 0 through 9.

JOB statement parameters.

The CLASS parameter.

Jobs are site-specific. You can check with your operations department about the job classes that are available for your use.

A good balance of job class assignments helps to make the most efficient use possible of the system.



63

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

You can find the job CLASS definition from SYSVIEW Primary Option Menu:

- 3 MENU JES2 JES2 job and output management
- 9 MENU JES2JOBS Job management
- 5 JOBCLASS Job classes

JOB statement parameters.

The NOTIFY parameter.

The NOTIFY parameter indicates the TSO/E user the system must notify upon job completion. If you use the NOTIFY parameter to specify your TSO/E user ID, the operating system automatically sends you a job completion message when your job ends.

For example, to have the system send a message to JSMITH when the job EX completes, you would code the NOTIFY parameter on a JOB statement as:

```
//EX      JOB    ...,NOTIFY=JSMITH
```

JOB statement parameters.

Are we on track?

Code a NOTIFY parameter that tells the system the TSO/E user is SDJONES

//JOB1 JOB 255,SMITH,_____

65

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is NOTIFY=SDJONES

JOB statement parameters.

Are we on track?

To whom does the NOTIFY parameter send a job completion message when the job ends?

- A. The MVS system administrator.**
- B. The specified JCL programmer.**
- C. A specified TSO/E user.**

The correct answer is C.

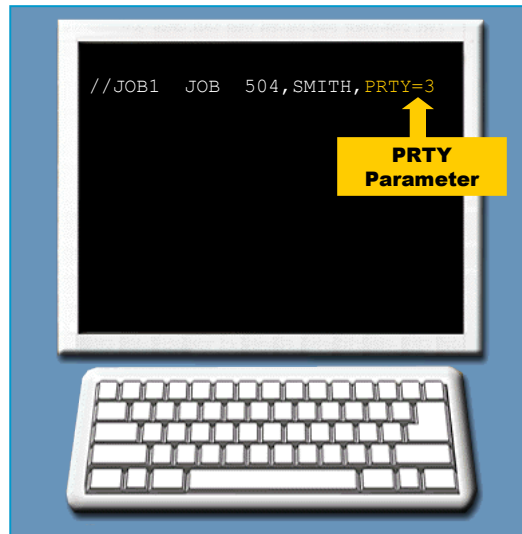
JOB statement parameters.

The PRTY parameter.

The PRTY parameter specifies a job's priority for selection within its job class. Usually, this parameter is meant to designate one job for execution over others in a class.

The range of PRTY values is usually 0 through 15, with 0 having the lowest priority.

When no priority has been specified, the system processes jobs within the same class in a first-in, first-out manner.



67

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

For example: If you have to give a specific priority of 3 to a job (within its default class), you will code the PRTY parameter on a JOB statement as has been shown here.

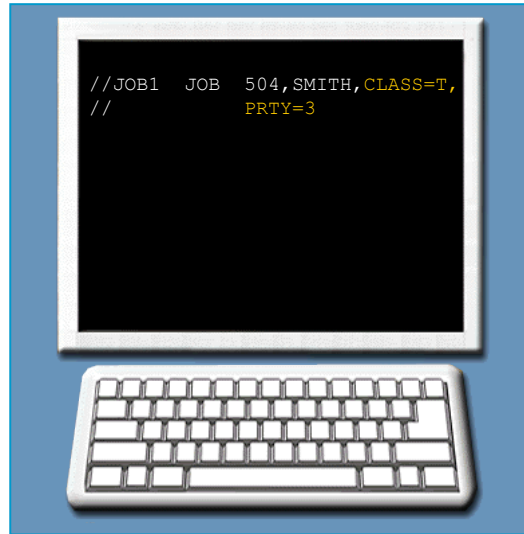
JOB statement parameters.

Coding CLASS and PRTY parameters.

If you have to give a specific priority to a job within a specific class, you will code both the CLASS and the PRTY parameters on the JOB statement.

For example: The JOB statement shown here specifies that the job has to run in class T and it has been accorded a priority of 3.

It is not significant in which order the CLASS and PRTY keyword parameters appear.



68

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

JOB statement parameters.

Are we on track?

Code a PRTY parameter that gives the job a specific priority of 5 (within its default class).

//JOB1 JOB 255,SMITH,_____

69

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is PRTY=5

JOB statement parameters.

The REGION parameter.

The REGION parameter specifies the amount of storage space (in kilobytes or megabytes) that has to be allocated to a particular job.

You can make use of this parameter to override the default region size set at your installation.

In case of a JOB statement, the region specified in the REGION parameter applies to all steps in a job and it overrides any REGION parameter coded on an EXEC statement.

```
//EXEC      JOB      ...,REGION=valueK  
//EXEC      JOB      ...,REGION=valueM
```

JOB statement parameters.

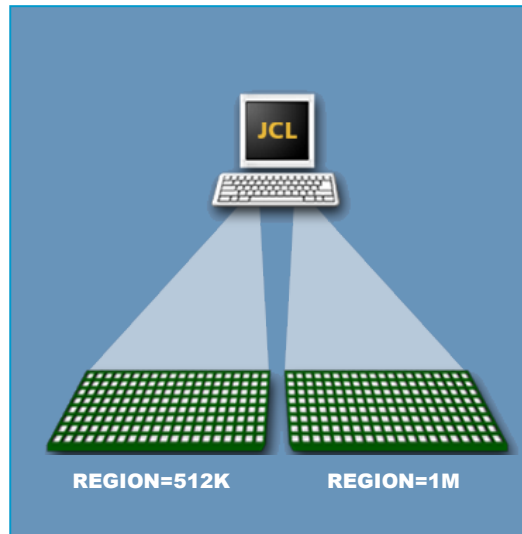
The REGION parameter.

In order to limit the virtual storage space of a job to 512 KB, you will need to code the REGION parameter on the JOB statement as:

```
//EX JOB ...,REGION=512K
```

In a similar manner, in order to limit a job's virtual storage space to 1024 KB bytes or 1MB, you will need to code the REGION parameter as:

```
//EX JOB ...,REGION=1M
```



Normally, when the REGION parameter makes a GETMAIN request or a dynamic request for more storage, it limits the amount of virtual storage available to a program.

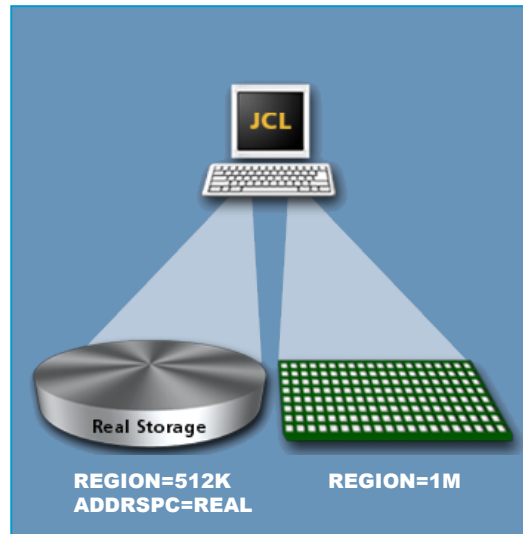
JOB statement parameters.

REGION and ADDRSPC parameters.

When you use the REGION parameter in conjunction with ADDRSPC=REAL, REGION specifies the amount of real storage.

For instance, to limit a job's real storage space to 512 KB, you code the REGION parameter, along with the ADDRSPC parameter on the JOB statement as:

```
//EX JOB . . . ,REGION=512K,  
// ADDRSPC=REAL
```



JOB statement parameters.

Are we on track?

Code a REGION parameter that limits a job's real storage space to 225K.

//JOB1 JOB 255,SMITH,_____

73

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is REGION=225K.

But you have to code ADDRSPC=REAL as well.

JOB statement parameters.

The TIME parameter.

The TIME parameter specifies a maximum amount of processor time available for the job. If the limit in the TIME parameter is reached, the job will terminate abnormally.

The syntax for the TIME parameter is:

```
//jobname JOB ...,TIME=(minutes,seconds)
```

The TIME parameter preserves processor time in case of an undetected error (like an endless loop) that may surface only during execution of program.

For example, to limit the CPU execution time to 2 minutes and 45 seconds, you will need to code the TIME parameter on a JOB statement as:

```
//EXAMPLE1 JOB 776,STUDENT,TIME=(2,45)
```

JOB statement parameters.

TIME subparameters.

Apart from the minutes and seconds subparameters, you can code three other subparameters with the TIME parameter.

These additional subparameters include

- **1440**
- **NOLIMIT**
- **MAXIMUM**

The 1440 indicates that a job is allowed to run for an unlimited amount of time. Literally, the subparameter 1440 means 1,440 minutes (i.e. 24 hours). However, the 1440 subparameter has special meaning to the operating system.

For instance, in order to allow your job to run indefinitely, you will code a JOB statement using the NOLIMIT subparameter as:

```
//EXAMPLE JOB 776,STUDENT,TIME=1440
```

75

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

JOB statement parameters.

TIME subparameters.

The **NOLIMIT** subparameter is identical in function to the **1440** subparameter.

Coding **NOLIMIT** with the **TIME** parameter will lead to the associated job running for an unlimited amount of time.

For instance, in order to allow a job to run indefinitely, you will have to code a **JOB** statement using the **NOLIMIT** subparameter as:

```
//EXAMPLE JOB 776,STUDENT,TIME=NOLIMIT
```

The **MAXIMUM** subparameter indicates that the associated job can run for **357,912** minutes, which is the maximum time the operating systems allows for a job (other than unlimited).

For example to limit the job's CPU processing time to **357,912** minutes, you would the code a **JOB** statement as:

```
//EXAMPLE JOB 776,STUDENT,TIME=MAXIMUM
```

JOB statement parameters.

Are we on track?

Code a TIME parameter that limits CPU execution time to 1 minute and 57 seconds.

//JOB1 JOB 255,SMITH,_____

77

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is TIME=(1,57)

JOB statement parameters.

The TYPRUN parameter.

The TYPRUN parameter identifies jobs that have special processing requirements.

The subparameters that can be used with the TYPRUN keyword are as follows:

- **COPY (for JES2 only) - This is used to tell the system to copy the input to a SYSOUT data set for output processing, but not to execute it.**
- **HOLD - this is used to tell the system to hold the job prior to execution, until the operator releases the job.**
- **JCLHOLD (for JES2 only) - This is used to tell the system to hold the job before completing the JCL processing. JES2 holds the job until the operator releases it.**
- **SCAN - This is used to tell the tell the system to scan the JCL for 78 syntax errors, but not to execute it.**

Try to enter !JCK command of CA JCL Check product on command line.

JOB statement parameters.

Are we on track?

Code a TYPRUN parameter that will check the JCL for syntax errors without actually executing the job.

```
//JOB1   JOB   255,SMITH,_____
//STEP1  EXEC  PGM=PROGRAMA
//DD1    DD   DSN=INPUT
```

79

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The correct answer is TYPRUN=SCAN

JOB statement parameters.

The USER parameter.

The **USER** parameter identifies the user ID of the person who submitted the job.

This parameter uses a **USERID** subparameter which must be 1 to 8 alphanumeric characters or national symbols. The first character cannot be numeric.

Many system facilities, including the Resource Access Control Facility (RACF) and the System Resource Manager (SRM) use the **USERID** subparameter.

For example, in order to specify a user ID named **HARRIS**, you will have to code the **USER** parameter as:

```
//EXAMPLE JOB 776,STUDENT,USER=HARRIS
```

Whether the **USER** parameter is required by you or not, depends on the requirements of your site. In most cases, the **USER** parameter is used in conjunction with the **PASSWORD** parameter.

In CA, **USER** and **PASSWORD** parameters are automatically added by the system:

```
//KOTMCOND JOB  
(127300030),'KOTMI01',NOTIFY=&SYSUID,MSGCLASS=A,CLASS=M  
// MSGLEVEL=(1,1),REGION=0M, YPRUN=SCAN  
  
// USER=KOTMI01,PASSWORD=
```

JOB statement parameters.

The PASSWORD parameter.

The **PASSWORD** parameter identifies a current RACF password for a job.

The **PASSWORD** parameter uses a job-specific subparameter, which can be 1 to 8 alphanumeric characters or national symbols.

Just as in the case of the **USER** parameter, the **PASSWORD** parameter may or may not be required at your site.

//EXAMPLE JOB ...,PASSWORD=password

To specify a password of **XYZ123**, with a user ID of **HARRIS**, you would code the **PASSWORD** and **USER** parameters as shown here.

**//EXAMPLE JOB 776,STUDENT,
// PASSWORD=XYZ123,USER=HARRIS**

JOB statement parameters.

Are we on track?

Complete the JOB statement by specifying a password of 123 and a user ID as JOHN.

//JOB1 JOB 255,SMITH,_____

82

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

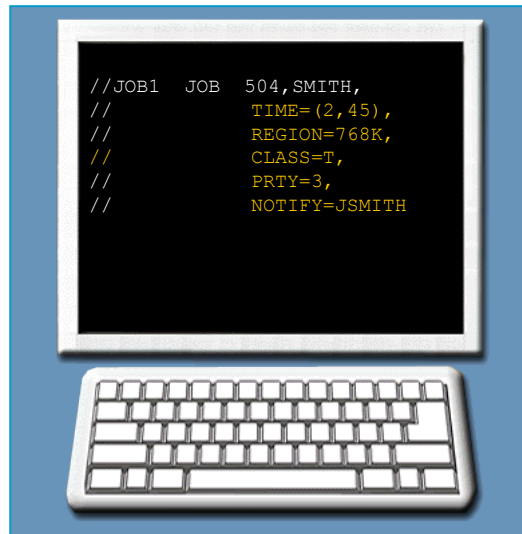
The correct answer is USER=JOHN,PASSWORD=123

JOB statement parameters.

Coding multiple keyword parameters.

You can code multiple keyword parameters on a single JOB statement in any order that you choose after any positional parameters are coded.

Here, the JOB1 job is limited in processing time to 2 minutes, 45 seconds, and has been allocated a limited space of 768 KB. This job has a job priority of 3 within class T. This means that job named JOB1 will run ahead of all other jobs in class T that have a lower priority value (PRTY value) than 3.



83

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

The operating system notifies user JSMITH, upon the complete execution of the job.

JOB statement parameters.

Glossary.

Virtual Storage

Storage space that may be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses.

TSO/E

Time Sharing Option Extensions. In the MVS/ESA environment, TSO/E provides virtual storage constraint relief.

RACF

Resource Access Control Facility. An IBM licensed program that provides for access control by identifying and verifying the users of the system, by authorizing access to protected resources, by logging the detected unauthorized attempts to enter the system, and by logging the detected accesses to protected resources.

84

Copyright © 2006 CA. All trademarks, trade names, services marks and logos referenced herein belong to their respective companies.

See z/OS Basics training to remember that we use CA's Top Secret security product instead of the IBM's RACF.

JOB statement parameters.

Unit summary.

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

- **Explain the purpose and syntax of the JOB statement.**
- **Define the JOB name.**
- **Code positional parameters and keyword parameters.**
- **Code the most commonly used keyword parameters on a JOB statement.**
- **Code additional JOB statement parameters.**