

## **6.- ARCHIVOS POR LOTES.**

En la interfaz de comandos se puede trabajar de dos maneras:

1.- Modo interactivo: En este caso el usuario introduce una orden, el sistema operativo la ejecuta y produce una respuesta, a continuación si usuario introduce otra orden, el sistema operativo la ejecuta y produce una respuesta y así sucesivamente.

2.-Modo batch.- El sistema operativo ejecuta varias órdenes seguidas sin intervención del usuario.

Un archivo por lotes es un archivo de texto ASCII en el que se escriben en líneas sucesivas varias órdenes de la interfaz de comandos, tal y como se escribirán en la línea de órdenes, a este archivo se le da una extensión .BAT y al introducir su nombre en el prompt se ejecutan automáticamente todas las órdenes escritas sin intervención del usuario.

Todos los archivos por lotes han de tener la extensión .bat, estos archivos se crean con un editor o procesador de textos ASCII, como el edit, o en Windows el bloc de notas; también se podría utilizar el Word o el Word perfect pero a la hora de guardarlo había que seleccionar el formato sólo texto, se ejecutan escribiendo su nombre sin extensión en la línea de comandos y se puede interrumpir su ejecución pulsando Ctrl+C.

Si bien cualquier comando que se pueda ejecutar en la línea de órdenes puede formar parte de un archivo por lotes, hay diez órdenes que suelen utilizarse en los archivos por lotes para mejorar su rendimiento: @, CALL, CHOICE, ECHO, FOR, GOTO, IF, PAUSE, REM, SHIFT.

Además pueden utilizarse los siguientes símbolos:

:NOMBRE → Etiqueta.

%NÚMERO → Parámetros sustituible.

%NOMBRE VARIABLE% → Variable sustituible.

%%LETRA → Variable de la orden FOR.

Según la función que realizan estás órdenes y símbolos internos a los archivos por lotes pueden agruparse de la siguiente manera:

1.- Gestión de mensajes: @, REM, ECHO.

2.- Gestión de parámetros: SHIFT, %NÚMERO.

3.- Gestión de la ejecución: CALL, CHOICE, PAUSE, IF, FOR, GOTO, :NOMBRE.

4.- Gestión de las variables de entorno: %NOMBRE VARIABLE%

## **GESTIÓN DE MENSAJES.**

-El ECO es el efecto por el cual se visualiza en pantalla las órdenes que se introducen en la línea de comandos.

La orden @ sirve para desactivar el eco de una orden, para ello se escribe la @ delante de la orden: "c:\>@cd c:\dos" → ¡ERROR! sólo funciona dentro de los archivos por lotes.

La orden ECHO OFF desactiva el eco de todas las órdenes que se escriban a continuación en un archivo por lotes, la orden ECHO ON vuelve a activar el eco, las órdenes siguientes se vuelven a ver en pantalla, si escribimos solamente ECHO, nos devuelve el estado del eco, si esta en on o en off.

La orden ECHO mensaje, visualiza en pantalla el mensaje aunque el eco esté desactivado, con las siguientes salvedades: si dentro del mensaje si

queremos que se escriban los caracteres de redireccionamiento, y la pipe tendremos que escribirlos entre comillas, análogamente si deseamos que aparezca el % tendremos que escribir %%.

La orden ECHO no afecta a los mensajes de salida que producen las diversas órdenes de la interfaz de comandos, si aparece un error, eso si aparecería en la pantalla, para evitar esto lo que se hace es redireccionar las órdenes al dispositivo NULL.

"Orden >NULL" esto no funciona con las órdenes DEL y ERASE pues si no existe el archivo que se quiere borrar siempre aparecerá el mensaje "Archivo no encontrado", para evitar que aparezcan estos mensajes siempre que utilicemos DEL o ERASE lo haremos de la siguiente manera: "IF Exist archivo DEL(ERASE) archivo".

*Notas:*

1.-Si en la primera línea de un archivo autoexec.bat escribimos la orden echo off todas las siguientes órdenes no se verán en pantalla pero si se verá la propia orden echo off, para evitar esto escribiremos @echo off, y de esta manera no será nada en la pantalla.

2.-Al terminar de ejecutarse un archivo por lotes el ECO se activa automáticamente.

3.-Si se escribe echo off en la línea de comandos desaparece el prompt, para que vuelva a aparecer hay que escribir echo on.

-La orden REM sirve para introducir comentarios internos, su sintaxis es: "REM[comentarios]" , la orden REM sin comentarios se utiliza para dejar líneas en blanco, pero para esto es mejor utilizar ECHO. .

Si el ECO esta activo, se visualiza en pantalla tanto REM como los comentarios y si el eco está desactiva no se visualiza REM ni los comentarios, no obstante se procesa y esto implica una pérdida de tiempo, para evitar esto, es mejor introducir los comentarios como se fuesen una etiqueta es decir con : delante.

La orden REM se utiliza sobre todo al depurar archivos por lotes, pues hay que evitar que dentro de los primeros caracteres de los comentarios no coincidan con una etiqueta llamada por goto.

La orden REM se escribe delante de otra orden para que esta se interprete como un comentario y no se ejecute.

La orden ECHO . introduce una línea en blanco a la salida; si escribimos sólo REM y el eco está activo, permite introducir una línea en blanco a la salida, otra solución es escribir : Comentarios.

Otra manera de depurar programas es mediante la orden COMMAND, siendo su sintaxis "COMMAND /y /c" nombre\_del\_archivo\_por\_lotes", el /y hace que se ejecute el archivo paso a paso y el /c hace que se ejecute el archivo por lotes.

### **Ejemplo de creación de un archivo por lotes:**

```
C:\> copy con básico.bat
@echo off
rem se personaliza el prompt
prompt $e[44m$P$g
: se solicita la fecha y la hora → comentario
echo introduzca la fecha → mensaje al usuario.
```

```

Echo. → línea en blanco en la salida.
date
echo introduzca la hora → mensaje al usuario
time
: se presenta la versión del sistema operativo → comentario.
Ver
ctrl.+c
C:\>

```

Se ejecuta el archivo por lotes escribiendo c:\>básico y pulsando ENTER o bien c:\>command /y /c básico

## GESTIÓN DE PARÁMETROS.

Al llamar a un archivo por lotes se le pueden pasar parámetros que pueden variar de una llamada a otra.

Para crear un archivo por lotes que se puedan aplicar a datos distintos cada vez que se ejecuta el archivo por lotes se introducen los parámetros sustituibles (% número), dentro del archivo por lotes en donde número es un número decimal, entero que puede valer de 0 a 9.

### **Ejemplo.**

```

Echo -----
@echo off
rem Programa que muestra en pantalla el contenido
rem de tres archivo. Nombre del programa visual.bat
type %1 |more
type %2 |more
type %3 |more
echo-----

```

Para llamar a este programa escribiremos en el prompt:

```

C:\> Visual config.sys autoexec.bat command.com
           %1           %2           %3

```

Comentarios: Pueden darse las siguientes posibilidades.

1.- Que se introduzcan tres archivos (tres parámetros), en este caso el archivo por lotes se ejecuta correctamente visualizando los tres archivos como argumentos.

2.- Se introducen más de tres archivos, en este caso se ignoran los parámetros sobrantes y se ejecuta el archivo con los tres elementos primeros.

3.-Se introducen menos de tres archivos, los parámetros sustituibles que no reciben un archivo en las llamadas se sustituyen internamente por un espacio en blanco. Con el ejemplo tendríamos por lo tanto type (espacio en blanco) |more y se produciría un error.

Si se trata de otra orden que pueda funcionar sin ningún parámetro no se producirán errores y el archivo por lotes se ejecutará aparentemente sin ningún error.

*Nota: El %0 hace referencia a la trayectoria y nombre del archivo por lotes, tal y como se ha escrito en la línea de órdenes.*

Los símbolos desde el %1 al %9 sólo permiten identificar los nueve primeros parámetros que se escriban en la orden de llamada al archivo por lotes, para poder asignar mas de nueve parámetros se utiliza la orden SHIFT de la siguiente manera: se escribe SHIFT sólo en una línea del archivo por lotes, esto hace que cada vez que se ejecute la orden SHIFT se desplace una posición a la derecha el comienzo de la numeración de parámetros.

Antes	%1	%2	%3	%4	%5	%6	%7	%8	%9	
Lotes	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Después de Shift		%1	%2	%3	%4	%5	%6	%7	%8	%9

### **Ejemplo:**

```
@echo off
echo -----
rem Programa DEMO.BAT
echo programa %0
echo -----
echo.
echo llamada al programa
echo -----
echo.
echo %1 %2 %3
echo.
echo Salida del programa %0
echo-----
echo El primer parámetro introducido es %1
SHIFT
echo El segundo parámetro introducido es %1
SHIFT
echo El tercer parámetro introducido es %1
echo-----
^C
```

### **¿CÓMO CORREGIR EL FALLO DE INICIO DE WINDOWS 95?**

AL instalar win 95 OSR2 veremos que aparecerá un mensaje que dice “Windows está inicializando el dispositivo IOS: error de protección de Windows debe reiniciar el ordenador”

Este es un error que sólo se presenta en la versión OSR2 de Windows 95.

#### **Consejos de Microsoft:**

1.- Iniciar Windows en el símbolo del sistema y modificar los archivos config.sys y autoexec.bat de manera que no se haga ninguna referencia al archivo SMARTDRV.exe y reiniciar el sistema.

2.- El problema sólo se produce en determinadas configuraciones del sistema y no se da si esta instalado en ele equipo el Internet Explorer 4.0

3.-Esto se produce en sistemas que tienen 32Mb de RAM y un gran número de controladores de dispositivo.

4.-Este problema puede producirse si un virus a dañado la cadena INT 21 y INT 13 (servicios de DOS y BIOS) antes de que cargue Windows o bien si se ha detectado un controlador de dispositivos no seguro o si falta o está dañado un controlador en modo protegido en el archivo IOSURSYS ubicado en la carpeta Windows\System\IOSUBSYS.

Esto puede ocurrir si se ha instalado software o hardware nuevos, de tal manera que el proceso de instalación de estos se halla realizado alguna acción que cause el problema actual. También puede producirse si se ha quitado software hardware anticuados.

Recomendación: Vuelva a la configuración original y observar si ya no aparece el error, en este caso el problema es que la BIOS está anticuada y hay que actualizarla.

5.- si se sigue produciendo el error, intente iniciar el equipo a prueba de errores, si no puede, entonces se hace lo siguiente: intentamos arrancar el equipo en el modo de confirmación paso a paso, para crear un archivo bootlog.txt, en el método paso a paso le vamos diciendo que ejecute orden a orden, hasta que se corta, pero se ha creado un archivo que abriremos con el edit. Si alguno falla:

1.- Toma nota del nombre del controlado que falla, que no se a cargado con éxito (load failure)

2.- Ir al directorio Windows\system\iosubsys, y buscar los archivos con el mismo nombre de que falta y se le cambia el nombre de .VXD a .OLD y reiniciar el sistema. Windows 95 usa una serie de controladores fundamentales, que no se cambian:

Biosxla.vxd	Vmcpv.vxd
Configmg.vxd	Vtdapi.vxd
Dynapage.vxd	Vcache.vxd
Dosmgr.vxd	Vdd.vxd
Ebios.vxd	Vkd.vxd
Ifsmgr.vxd	Vtd.vxd
Int13.vxd	Vxdidr.vxd
Ios.vxd	Vcd.vxd
Pageswap.vxd	Vcound.vxd
Shell.vxd	Vfat.vxd
V86mmgr.vxd	Vpicp.vxd
Vcomm.vxd	Vwin32.vxd
Vdmap.vxd	

Se recomienda instalar Windows en una carpeta nueva para comprobar si es compatible con el nuevo hardware instalado.

6.-Se puede producir el error cuando la memoria RAM por debajo de 1Mb se ha llenado completamente antes de inicializar el controlador IOS.vxd, por que este controlador necesita utilizar memoria por debajo de 1Mb para poder utilizarla como buffer en modo real. La solución seria ajustar el tamaño máximo de caché que utiliza el controlador Vcache.vxd que por defecto utiliza 9Mb, ha que hacer que este controlador asigne menos memoria.

- se edita el archivo System.ini

- Debajo de [Vcache] se escribe: maxfilecache=6144.

7.- Otra posible causa `puede ser la siguiente: En el prompt escribimos mem /c /p, y nos dice que controladores están cargados en memoria; el HIMEN

tiene un tamaño de 43Kb y debe de valer mas o menos 1Kb. Se corrige con un parche.

8.- Otra causa es que en procesadores superiores al AMD K6 2 a 350 MHz se modificó la arquitectura interna del procesador, para solucionar este problema se hace con otro parche.

### **CONTROL DE LA EJECUCIÓN.**

Por defecto la ejecución de los archivos por lotes es secuencial, para poder alterar la ejecución secuencial existe una serie de órdenes, que son las siguientes:

1.-PAUSE: detiene la ejecución del archivo por lotes hasta que se pulsa una tecla.

2.- GOTO etiqueta: provoca un salto incondicional a la orden siguiente a la fila en la que se encuentra la etiqueta.

3.-IF: si se cumple una cierta condición se ejecutan ciertas órdenes y si no se cumple la condición no se ejecutan.

4.- FOR: Repite un conjunto de órdenes un número determinado de veces.

5.- CALL: llamada a otro archivo por lotes

6.- CHOICE: permite escoger una opción entre varias, y dependiendo de la opción elegida devuelve un código de salida.

7.- :NOMBRE: una etiqueta, una sucesión de caracteres que puede tener una longitud máxima de 127 caracteres, pero el sistema operativo sólo considera los ocho primeros caracteres, es decir, que hay que evitar etiquetas que tengan iguales los ocho primeros caracteres, pues de existir la orden goto saltaría a la etiqueta colocada en primer lugar en el archivo por lotes.

### **ORDEN PAUSE.**

La sintaxis es *PAUSE [mensaje]*, el mensaje en general se utiliza para indicar al usuario el motivo de la pausa, por ejemplo “pulse una tecla para continuar”. La orden PAUSE produce los siguientes efectos:

1.- Si el eco está activado se visualiza el mensaje, si no no.

2.- se detiene la ejecución del archivo por lotes hasta que el usuario pulse una tecla.

3.- Si no se especifica un mensaje aparece “pulse una tecla para continuar”.

Para hacer que aparezca el mensaje si el eco está activado pueden utilizarse dos técnicas:

1.-

```
.  
.   
.   
.   
.   
@echo on  
@pause [mensaje]  
@echo off  
.
```

2.-

```
.  
.   
.   
echo mensaje  
pause > nul  
.  
.
```

### Ejemplo:

```
@echo off  
: programa que solicita al usuario que introduzca un disquete en la  
: unidad a: y luego presente el directorio de a:.  
: se llama VERA.bat  
echo introduzca un disquete en la unidad a: y pulse ENTER  
pause > nul  
dir a:*.*
```

### ORDEN GOTO.

La sintaxis será: *GOTO etiqueta*

```
.  
.   
etiqueta  
orden
```

Hace un salto incondicional desde el GOTO hasta la etiqueta con el mismo nombre especificado detrás de la orden GOTO.

Si no existe una etiqueta que coincida con la especificada en la orden GOTO se muestran un mensaje que dice “etiqueta no encontrada”.

**Ejercicio:** Hacer un archivo por lotes que permita copiar varios disquetes desde la unidad a : al directorio actual.

```
@echo off  
: rem programa Copias.bat  
CLS  
echo -----  
echo --- Este programa copia los archivos de varios disquetes ---  
echo --- desde la unidad a: al directorio actual ---  
echo -----  
echo.  
echo introduzca un disquete en la unidad a: y pulse enter  
pause > nul  
copy a:\*.*  
: comienzo  
echo.  
echo .....  
echo.  
echo Si desea copiar otro disquete , introdúzcalo en a: y pulse ENTER  
echo en caso contrario pulse ctrl.+c.
```

```
pause > nul
copy a:\*.*
goto comienzo
```

## **ORDEN IF**

Permite Realizar un salto condicional en la secuencia del programa, se evalúa una condición y si es verdadera se ejecuta una orden especificada o se realiza un salto. Puede tener 6 formatos:

- 1 a) IF EXIST archivo orden\_a\_ejecutar
- 1 b) IF NOTEXIST archivo orden\_a\_ejecutar
- 2 a) IF ERRORLEVEL número orden a ejecutar
- 2 b) IF NOT ERRORLEVEL número orden\_a\_ejecutar
- 3 a) IF CADENA1==CADENA2 orden\_a\_ejecutar
- 4 b) IF NOT CADENA1==CADENA2 orden\_a\_ejecutar

Archivo es la trayectoria y nombre de un archivo.

Orden a ejecutar sólo puede ser una única orden válida del sistema operativo.

Si se quieren ejecutar varias órdenes existen dos procedimientos:

- a) poner una orden GOTO :etiqueta
- b) Poner una orden CALL archivo.bat.

### **Formatos 1 A y 1 B.**

IF EXIST archivo orden\_a\_ejecutar; si existe el archivo especificado se ejecuta la orden, y si no existe se continúa en la línea siguiente.

IF NOT EXIST archivo orden\_a\_ejecutar; si no existe el archivo se ejecuta la orden y si existe no se ejecuta.

Este formato suele utilizarse para identificar un disquete, se crea un archivo en el disquete que pueda identificarlo, por ejemplo disk5.ID y en el archivo por lotes se escribe algo parecido a lo siguiente:

```
IF NOT EXIST a:\disk5.id goto el
```

```
.
```

```
.
```

```
:el
```

```
echo introduzca el disco 5 en a:
```

Ejemplo borrar.bat

```
@echo off
```

```
if exist %1 del %1
```

```
if not exist %1 echo no se puede borrar un archivo que no existe.
```

### **Formatos 2 A y 2 B.**

A veces se necesita saber si una orden se ha ejecutado correctamente para ello se utilizan estos formatos. Esos formatos se basan en los códigos de salida generados por ciertas órdenes.

Un código de salida es un valor numérico que los programas devuelven al sistema operativo cuando terminan su ejecución, en general sirven para especificar la causa que ha procesado la terminación del programa. Los códigos de salida permanecen accesibles a la orden IF ERROR LEVEL hasta que se ejecuta otro programa o una orden externa de la interfaz de comandos.



Siempre que se ejecute una orden que necesita cargar en memoria un archivo, se pierde el valor actual del código de salida, asumiendo el nuevo código que devuelve el programa recién ejecutado.

Las únicas órdenes que no alteran el código de salida son las órdenes internas (las almacenadas en el command.com) al menos hay siete órdenes externas que devuelven código de salida y que por lo tanto se pueden tratar con IF ERROR LEVEL, estas son: Backup, Diskcomp, Format, Replace, Restore, Xcopy.

Siendo los códigos de salida de la orden Xcopy:

0 → Los archivos fueron copiados sin error.

1 → No se encuentran los archivos para copiar.

2 → El usuario pulsó Ctrl.+c para cancelar xcopy.

4 → Ocurrió un error interno. No hay suficiente memoria o espacio en el disco, se introdujo un nombre de unidad no válido o se utilizó una sintaxis incorrecta en la línea de órdenes.

5 → Cuando se produce un error de escritura en el disco.

Un problema al manejar ERRORLEVEL es que la condición IF ERROR LEVEL n es verdadera si el código de salida es mayor que n, es decir: si la condición es IF ERROR LEVEL n orden se ejecuta la orden si el código de salida es mayor o igual que n.

Si la condición es IF NOT ERRORLEVEL n orden se ejecuta la orden si el código de salida es menor que n. Para manejar esta situación existen dos técnicas:

1.- Al procesar los códigos de salida se empieza por el mas alto: If errorlevel n orden; if not errorlevel n-1 orden.

2.- Cuando sea necesario que la orden sólo se ejecute para un único valor de errorlevel se hace lo siguiente:

If error level n1 if not errorlevel n1+1 orden; es decir, el código de salida en n1 o superior pero inferior a n1+1 se ejecuta la orden, lo cual es equivalente a que la orden sólo se ejecuta si el código de salida es igual a n1.

Ejemplo:

```
@echo off
```

```
:rem programa copias2.bat
```

```
xcopy c:\*.* /e
```

```
if errorlevel 1 goto error
```

```
if error level 0 echo los archivos se han copiados correctamente
```

```
goto fin
```

```
:error
```

```
echo se produjo un error durante la copia
```

```
:fin
```

### Formatos 3 A y 3 B.

En este formato se comparan dos cadenas, aquí se distingue entre mayúsculas y minúsculas, sin embargo no se tienen en cuenta los espacios en blanco.

Cuando se ordena ejecutar un archivo por lotes puede ocurrir que nos olvidemos de escribir los parámetros que puedan existir en el archivo por lotes, en este caso el sistema operativo los sustituye por espacios en blanco, con lo que el comportamiento del archivo por lotes es impredecible, por este motivo, es necesario conocer si se han introducido parámetros al llamar a un archivo por lotes. Para ello, se utiliza la comparación de cadenas:

IF "%1"==" " → es verdadera si no se han introducido parámetros. Es falsa si se han introducido parámetros (no se consideran los espacios en blanco).

IF a%1==a

Ejemplo:

@echo off

:rem Programa borrar.bat

If a%1==a echo debe introducir al menos el nombre de un archivo

If not a%1==a del %1

If not a%2==a del %2

### **ORDEN CALL.**

Se utiliza para llamar a un segundo archivo por lotes, de tal manera que finalizadas las órdenes de este segundo archivo se regresa al archivo que hizo la llamada y se ejecuta la orden existente en la línea siguiente a la línea en que se encuentra la orden CALL.

CALL archivo\_por\_lotes2 [parámetros2º archivo]

Ejemplo:

@echo off

:rem programa llamada1.bat

echo línea1

call llamada2

echo línea4

echo línea5

@echo off

:rem programa llamada 2.bat

echo línea2

echo línea3

### **ORDEN CHOICE.**

Esta orden sólo puede utilizarse dentro de un archivo por lotes, permite que el usuario escoja una opción (tecla) entre varias.

Dependiendo de la opción elegida devuelve un código de salida que puede manejarse con ERRORLEVEL, la sintaxis es:

CHOICE [mensaje] [/c[:]teclas] [/n] [/s] [t[:] c,nn].

En donde:

Mensaje significa el mensaje que se desea mostrar al usuario, si tenemos que incluir \ o un modificador se escribirán entro comillas; si no se especifica un mensaje se muestra el símbolo del sistema.

/c[:] especifica las opciones (teclas) posibles, que puedan elegir el usuario. Si el usuario pulsa la primera opción devuelve el código de salida 1 y si pulsa la segunda opción se devuelve 2 y así sucesivamente.

Si el usuario pulsa una tecla no asignada se emite un sonido, es decir se envía in comando bel o 47h a la consola.

Si no se especifica este parámetro se asumen las opciones por defecto que son /s y /n.

Si choice detecta una secuencia de error devuelve un error para errorlevel de 255, si el usuario pulsa ctrl.+c devuelve el valor 0

Ejemplo: Choice /c:snc→ el usuario verá→ [S,N,C]?

Ejemplo: Choice Si No o Continuar→Si No o Continuar [S,N,C]?

/N Evita que choice presente el prompt. El usuario sólo verá el mensaje especificado y las teclas especificadas serán válidas.

Ejemplo: Choice ¿Si No o Continuar? /n → ¿Si No o Continuar?

/S cuando se especifica se distingue entre mayúsculas y minúsculas, si no se especifica no se distingue.

/T[:]c, nn] Se realiza una pausa de nn segundos (entre 0 y 99); si transcurrido estos segundos el usuario no ha realizado ninguna selección se ejecutara la opción c especificada.

Ejemplo: choice /c:sn /t:n,5; si después de cinco segundos no ha pulsado una tecla choice seleccionará la tecla n y se devolverá un errorlevel 2. si el usuario presiona una tecla antes de los 5 segundos se devuelve el valor correspondiente a dicha selección.

Ejemplo:

@echo off

:reiniciar

:ren programa menú.bat

cls

echo.

echo E edit

echo A antivirus

echo C copia de seguridad

echo D defragmentador de c:

echo S salir

echo.

choice elija una opción /c:EACDS /N /TS,15

if error level 5 goto e5

if errorlevel 4 if not errorlevel 5 goto e4

if errorlevel 3 if not errorlevel 4 goto e3

if errorlevel 2 if not errorlevel 3 goto e2

if errorlevel 1 if not errorlevel2 goto e1

:e1

edit

goto fin

:e2

c:\dos\msav

goto fin

:e3

c:\dos\msbackup

goto fin

:e4

c:\dos\defrag

goto fin

:fin

```
goto reiniciar
:e5
echo.
```

## **VARIABLES DE ENTORNO Y ARCHIVOS POR LOTES**

Desde un archivo por lotes se puede conocer a las variables de entorno, para ello se escribe %nombrevariable% que suele llamarse parámetro nombrado.

Los parámetros nombrados sustituyen el nombre de una variable por su valor, ejemplo: SET nb=command.com → se crea la variable de entorno nb.

Si nos vamos al directorio raíz cd/ entonces escribimos IF EXIST %nb% dir es equivalente a IF EXIST command.com dir

Ejemplo:

```
@echo off
@echo el valor de la variable PATH es %PATH%
@echo el path actual es %PATH%
```

Ejemplo:

IF %temp%=="" Set temp=c:\temp; es decir si no se ha definido la variable de entorno temp se define y se le da el valor c:\temp.

## **ORDEN FOR.**

Realiza un mismo proceso un número determinado de veces, la sintaxis es : FOR %%letra In (lista de valores) Do orden\_a\_ejecutar. En donde:

Letra.- es una única letra que representa una variable que va tomando los valores establecidos en (lista de valores), cada vez que se realiza el proceso definido en orden\_a\_ejecutar.

(lista de valores).- es una lista que se escribirá entre paréntesis y los diversos valores estarán separados por espacios en blanco. Se pueden usar los caracteres comodines.

Orden\_a\_ejecutar.- ha de ser una sola orden que se repite tantas veces como elementos hay en lista de valores.

Ejemplo: For %%x in (na?.txt) do type %%x|more; esta orden seria equivalente a:

```
Type na1.txt|more
Type na2.txt|more
```

*Nota: la orden FOR también puede escribirse en la línea de comandos, sustituyendo el %%letra por %letra, por ejemplo: c:\>FOR %x in (\*.\*) do chkdsk %x.*

Ejemplo:

```
For %%c In (pepe.txt maria.txt jose.txt *.dat) Do type %%c
```

### **Poder repetir varias ordenes:**

Escribiremos: FOR %%letra In (L.valores) Dos Call archivo\_por\_lotes %%letra.

Para poder repetir varias órdenes se escribirán en un archivo por lotes independiente, de esta manera la orden FOR no realiza repetitivamente una sola orden, si no que repite todas las órdenes existentes en el otro archivo por lotes.

### **INSTALACIÓN DE WINDOWS 98 OEM SOBRE WINDOWS 95.**

Vamos al directorio donde tengamos guardado los archivos de Windows y busquemos el win.com y lo renombramos a win.old desde la interfaz de comandos de Windows 95.

C:\Windows95\win.com ó C:\Win95\win.com

Ren win.com win.old

### **ARCHIVOS CLAVE DE WINDOWS 98 CUANDO SE INSTALAN EN EL DISCO DURO.**

Archivo		Ubicación	Descripción
Io.sys	H	Directorio raíz unida de arranque	Sistema operativo en modo real y detección del sistema
Command.com		Directorio raíz u.a.	Interprete de comandos interfaz de comandos
Win.com		C:\windows	Parte en modo real para iniciar w98 interfaz gráfica
Vmm32.vxd		C:\windows\system	Administrador de máquinas virtuales en modo protegido
System.Dat	H	C:\windows	Registro de Windows
Rb0001.cab Hasta Rb005.cab		C:\windows\susbckup	Copia de seguridad del registro de Windows
System.new		Directorio raíz u.a	Registro del sistema creado al instalar la 1ª vez
User.dat	H	C:\windows	Registro del usuario
User.new		Directorio raíz u.a.	Registro del usuario creado por instalar la 1ª vez
Setuplog.txt	H	Directorio raíz u.a.	Anotaciones del proceso de instalación
Detlog.txt		Directorio raíz u.a	Anotaciones de la detección del hardware
Bootlog.txt		Directorio raíz u.a.	Anotaciones del proceso de inicio de win98
Protocol.ini		C:\windows	Archivo de inicialización del sistema
Msdos.sys	H	Directorio raíz u.a.	Archivo de configuración de inicio
System.1st	H	C:\windows	Registro creado final de instalar
System.ini		C:\windows	Archivo inicialización del sistema
Explorer.exe		C:\windows	Archivo ejecutable de la interfaz grafica de ordenes

Drvspace.bin		Directorio raíz u.a.	Soporte para la compresión de discos
--------------	--	----------------------	--------------------------------------

Si el disco está comprimido o tenía instalado previamente el ms-dos aparecen otros valores como el drvspace.ini el dlbspace.bin el msdos.--- el msdos.dos el msdos.sys.... el wina28.386..... .

---