PROGRAMACIÓN SHELL

Un shell es un programa, similar al COMMAND.COM, que interpreta las órdenes introducidas al sistema.
Igual que en el DOS, las órdenes pueden introducirse directamente desde la línea de comando o pueden realizarse programas que ejecuten de forma automática un conjunto de instrucciones.
Cada fichero programado o programa en Linux se denomina **guión** o **script**.
Los pasos a seguir para crear un guión son:

- Escribirlo utilizando un editor de texto (con el comando cat o con el editor **v**i)
- Asignarle el permiso de ejecución (**x**) con el comando chmod
- Dependiendo del shell en el que estemos, para ejecutarlo se utilizan distintos métodos. Nosotros utilizaremos el siguiente: escribir **./nombredelguion**

Comando Básicos

- **echo** .- Para mostrar mensajes por pantalla
- **#** .- Para añadir comentarios al guión. No se ven en pantalla.
- **read** .- Permiten aceptar variables por teclado. Esta orden detiene el flujo de ejecución del script a la espera de una entrada por teclado. El script continua cuando se pulsa Intro. También podemos finalizar con Ctrl+d que además detiene el script.

Variables

Para utilizar variables hay que asignarles valores. Dicha asignación puede hacerse utilizando el comando read, por asignación directa, etc.

Ej: echo "Introduzca su nombre"; read nombre
echo "Escribe su primer apellido"; read apel1
echo "Escribe su segundo apellido"; read apel2
echo "Tu nombre completo es:" $nombre $apel1 $apel2

Para recuperar el valor de una variable hay que utilizar el signo $ seguido del nombre de la variable.

Para asignarle una valor a una variable se escribe:
variable=valor (NO DEJAR ESPACIO EN BLANCO)
mimail=pepe@arrakis.es
midireccion="C/ del Oro 5, 3º B"

También se puede asignar a una variable el resultado de un comando. Ej. fecha= ´date´
fecha=´date +%d-%m%-%y´

Parámetros

Son similares a los del DOS pero en este se utilizaban %1, %2, etc., y en Linux se utiliza $1, $2.... $9. Pero además de estos 9 parámetros existen otros que devuelven unos valores específicos:

$0 .- Nombre del guión ejecutado
$# .- Número de argumentos introducidos.
$? .- Código generado por la última orden ejecutada. Similar a ErrorLevel del

DOS.
$* .- Conjunto de todos los parámetros (Ver ejemplo en For)
$$.- Numero de identificación del proceso.

Estructura de un script

comando1;comando2 .- se ejecutan los dos comandos uno después de otro pero de forma independiente.
comando1|comando2 .- el resultado de la primera orden es la entrada de la segunda.
comando1&&comando2 .- la segunda orden se ejecuta sólo si la primera se ha ejecutado con éxito.

Comparación y Comprobación

test .- Esta orden sirve para evaluar una expresión. Se puede chequear la existencia o no de un fichero, se pueden comparar cadenas, números enteros, longitud de cadenas, etc.

Los parámetros mas importantes para analizar ficheros son:
-**e** .- el fichero o cadena existe
-**b**.- fichero especial de bloque
-**c**.- fichero especial de caracteres
-**d**.- para comprobar directorio
-**f**.- para ficheros ordinarios
-**s**.- para saber si el tamaño de un fichero es superior a 0 bytes
-**w**.- para saber si tiene permiso de escritura
-**r**.- para permiso de lectura
-**x**.- para permiso de ejecución
-**h**.- para comprobar enlaces simbólicos.

Los parámetros más importantes para analizar cadenas de caracteres son:
-**z**.- cierto si la longitud de la cadena es 0
-**n**.- cierto si la longitud de la cadena es distinta de 0
cadena = cadena.- cierto si las cadenas son iguales *(con espacios)*
cadena != cadena.- cierto si las cadenas son distintas.

Los parámetros más importantes para comparar números enteros son:
-**eq**.- igual a
-**ne**.- distinto de
-**it**.- menor que
-**le**.- menor o igual que
-**gt**.- mayor que
-**ge**.- mayor o igual que

Los operadores lógicos son:
!.- negación
-**a** .- and o Y lógico
-**o**.- or u O lógico

Delante y detrás de los parámetros tiene que haber un espacio en blanco.

Estructuras condicionales y repetitivas

If

Formato1

```
If expresion                if [ -e $1 ]
Then                        then
        Ordenes                     cat $1
Fi                          fi
```

Formato2

```
If expresion                if [ cad1 = cad2 ]
Then                        then
        Ordenes                     echo "Son iguales"
Else                        else
        Ordenes                     echo "Son distintos"
Fi                          fi
```

Formato3

```
If expresion
Then
        Ordenes
        Elseif expresion
                Then
                        Ordenes
                Else
                Ordenes
                fi
else
        ordenes
fi
```

Case

```
Case variable in
        1) ordenes ;;           a|A)
        2) ordenes ;;           b|B)
        *) ordenes;
esac
```

While

```
While [     ] (while [ "resp" != "S" –a "$resp" != "s" ])
do
        Órdenes
done
```

Until

```
Until [         ]     (until [ "$resp" = "N" ])
do
        Órdenes
done
```

for

```
for variable in lista    (for num in $*
do                              do
        órdenes                         echo $LOGNAME
                                        echo $HOME
done                            done)
```

EJEMPLOS

```
echo "Escribe tu nombre"; read nombre
echo "Escribe primer apellido";read apel1
echo "Escribe segundo apellido";read apel2
echo "Tu nombre completo es: " $nombre $apel1 $apel2
```

test n1 –eq n2	**cierto si n1 y n2 son iguales**
test n1 –ne n2	**cierto si n1 y n2 son distintos**
test n1 –ge n2	**cierto si n1 es mayor o igual a n2**
test n1 –gt n2	**cierto si n1 es mayor que n2**
test n1 –le n2	**cierto si n1 es menor o igual a n2**
test n1 –lt n2	**cierto si n1 es menor que n2**

test –r fichero	**cierto si el fichero existe y tiene permiso de lectura**
test –w fichero	**cierto si el fichero existe y tiene permiso de escritura**
test –x fichero	**cierto si el fichero existe y tiene permiso de ejecución**
test –s fichero	**cierto si el fichero existe y no está vacío**
test –d fichero	**cierto si el fichero existe y es un directorio**
test –f fichero	**cierto si el fichero existe y es un fichero normal**
test –e fichero	**cierto si el fichero o cadena existe**
test –h fichero	**cierto si el fichero existe y es un enlace simbólico**
test –b fichero	**cierto si el fichero existe y es un fichero especial de bloques**
test –c fichero	**cierto si el fichero existe y es un fichero especial de caract**

TEST

```
mkdir $1
if test $? –eq 0
        then
                echo El directorio se ha creado con éxito
fi

directorio = 'pwd'
if test $directorio = $HOME
        then
                echo "Esta en el directorio adecuado"
fi

if test  -f fichero1 –a –f fichero2
        then
                echo "Ambos son ficheros normales"
fi
```

CADENAS

```
echo "Escribe algo ";read cad1
echo "Otra vez ";read cad2
clear
if [ $cad1 = $cad2 ]
        then
                echo "iguales"
        else
                echo "distintas"
fi
```

CADENAS CON NEGACION

```
echo "Escribe algo ";read cad1
echo "Otra vez ";read cad2
clear
if [ $cad1 != $cad2 ]
        then
                echo "distintas"
        else
                echo "iguales"
fi
```

CADENA VACIA

```
echo "Escribe algo"; read cad
if [ $cad ]
        then
                echo "Tiene algo"
        else
                echo "VACIA"
fi
```

MENU

```
clear
respuesta=" "
while [ "$respuesta" != "S" -a "$respuesta" != "s" ]
do
        echo "Visualizar un fichero"
        echo "Borrar un fichero"
        echo "Salir"
        echo "Elige una opcion [V,B,S]";read respuesta
        case $respuesta in
                V|v)  echo "Que fichero quiere ver ?";read nombrefich
                        cat $nombrefich;;
                B|b) echo "Que fichero quiere borrar?";read nombrefich
                        rm $nombrefich;;
                S|s) echo "Hasta luego";
        esac
done
```

MENU 2

```
clear
respuesta=0
while [ $respuesta -ne 3 ]
do
        echo "Visualizar un fichero"
        echo "Borrar un fichero"
        echo "Salir"
        echo "Elige una opcion ";read respuesta
        case $respuesta in
                1)  echo "Que fichero quiere ver ?";read nombrefich
                        cat $nombrefich;;
                2) echo "Que fichero quiere borrar?";read nombrefich
                        rm $nombrefich;;
                3) echo "Hasta luego";
        esac
done
```

FOR

```
clear
for usu in $*
# PARA TODOS LOS PARAMETROS QUE SE INTRODUZCAN
# EN ESTE CASO SON USUARIOS
  do
        echo $LOGNAME
        echo $HOME
done
```