U4 ADMINISTRACIÓN Y ASEGURAMIENTO DE LA INFORMACIÓN

ACTIVIDAD PRÁCTICA 14: Automatización (SCRIPTS)

24 DE ABRIL DE 2023

SISTEMAS INFORMÁTICOS
Sergio Cobo García

Contenido

Ejercicio 1

01	El comando test dispone de un largo número de funciones que permiten hacer comparaciones. Llene las tres tablas siguientes, respectivas a los tres tipos de operadores de comparación de test. Indique qué valor tomaría la variable ?, en función de los valores introducidos (recuerde que 0 es verdadero en Linux).

Comparación numérica	
test n1 -eq n2	? valdría 0 si n1 es igual que n2
test n1 -ge n2	? valdría 0 si n1 es mayor o igual que n2
test n1 -gt n2	? valdría 0 si n1 es mayor que n2
test n1 -le n2	? valdría 0 si n1 es menor o igual que n2
test n1 -lt n2	? valdría 0 si n1 es menor que n2
test n1 -ne n2	? valdría 0 si n1 no es igual que n2

Comparación de cadenas de texto	
test s1 = s2	? valdría 0 si s1 es igual que s2
test s1 != s2	? valdría 0 si s1 es no es igual que s2
test -n s1	? valdría 0 si s1 es diferente a 0
test -z s1	? valdría 0 si s1 es igual a 0

Comparación de archivos	
test -d f1	? valdría 0 si f1 existe y es un directorio
test -e f1	? valdría 0 si f1 existe
test -f f1	? valdría 0 si f1 existe y es un archivo regular
test -r f1	? valdría 0 si f1 existe y tiene permisos de lectura
test -s f1	? valdría 0 si f1 existe y el tamaño es mayor que 0
test -w f1	? valdría 0 si f1 existe y tiene permisos de escritura
test -x f1	? valdría 0 si f1 existe y tiene permisos de ejecución
test f1 -nt f2	? valdría 0 si la fecha de modificación de f1 es más reciente que la de f2
test f1 -ot f2	? valdría 0 si la fecha de modificación de f1 es más antigua que la de f2

Ejercicio 2

02	Explique el funcionamiento de las comillas de este ejercicio, y por qué se utilizan.

Echo “$comando”: ejecuta el valor de la variable comando, en este caso se lleva a cabo el comando ls.
Echo `$comando`: usa el resultado de la variable comando (ls).
Echo ‘$comando’: muestra literalmente el texto escrito entre las comillas simples.

Ejercicio 3

03	Indique justificadamente cuál es la función del siguiente shellscript, indicando cuál es el significado más lógico de los parámetros. Escribid un comando equivalente a todo este shellscript.

Se crea el archivo tmp. Seguidamente se inicia un bucle para que en cada archivo .txt busque la palabra "examen" y si la encuentra añade la línea que la contiene a tmp. A continuación, cuenta el número de líneas en el archivo tmp y lo imprime. Finalmente elimina el archivo tmp.

Un comando que hace lo mismo que el shellscript es el siguiente.

```
sergio@sergio-VirtualBox:~/Escritorio$ grep "examen" *.txt >> tmp
 && wc -l < tmp && rm tmp
1
```

Ejercicio 4

04	Escriba un shellscript que a partir de un parámetro numérico N, ordene alfabéticamente los logins de usuario del sistema y muestre los N últimos.

```
#!/bin/bash

# Verificar si el número de parámetros es correcto
if [ "$#" -ne 1 ]; then
  echo "Número de parámetros incorrecto"
  exit 1
fi

# Obtener la lista de nombres de usuario y ordenarla alfabéticamente
lista_usuarios=$(cut -d: -f1 /etc/passwd | sort)

# Contar el número de usuarios y restar el número de usuarios que se quieren mostrar
total_usuarios=$(echo "$lista_usuarios" | wc -l)
usuarios_mostrar=$(($total_usuarios - $1))

# Mostrar los últimos N usuarios
echo "$lista_usuarios" | tail -$usuarios_mostrar
```

```
sergio@sergio-VirtualBox:~/Escritorio$ ./ejemplo2.sh 37
systemd-resolve
systemd-timesync
tcpdump
tss
usbmux
uucp
uuidd
vboxadd
whoopsie
www-data
```

Ejercicio 5

05	Escriba un shellscript que indique si nos encontramos en los primeros o últimos seis meses del año. Tenga en cuenta que el sistema puede estar en cualquier idioma, por lo tanto, utilice los parámetros del comando "date" para obtener un valor válido para cualquier idioma.

```
#EJERCICIO 5
#!/bin/bash

#Obtener el mes actual
mes=`date +%m`;

#Compruebo si es mayor o menos a 6 el número de mes
if (test $mes -le 6)
then
      echo "Está en los primeros 6 meses";
else
      echo "Está en los últimos 6 meses";
fi
```

```
alumati@laD20203ubt2204:/media/a
CRIPTS/PRACTICA$ ./ejercicio5.sh
Está en los primeros 6 meses
```

Ejercicio 6

06	Escriba un shellscript simple que, a partir de un número indeterminado de argumentos, salude a cada uno de los argumentos pasados.

```
#EJERCICIO 6
#!/bin/bash

#Creo un array de nombres
nombres=('Sergio' 'Paco' 'Antonia' 'Maria' 'Anastasia' 'Rigoberta' 'Manolo')

#Imprimo la respuesta a cada uno de ellos
for i in ${nombres[@]}
do
      echo "Hola $i"
done;
```

```
Sergio@Sergio-PC
$ ./ejercicio6.sh
Hola Sergio
Hola Paco
Hola Antonia
Hola Maria
Hola Anastasia
Hola Rigoberta
Hola Manolo
```

Ejercicio 7

07	Escriba un shellscript que, a partir de un único parámetro N, y utilizando el bucle "while", muestre por pantalla una progresión aritmética de N términos (1, 2, 3, 4...) y una progresión geométrica de N términos (1, 2, 4, 8, 16...). El número de términos de las sucesiones será el primer parámetro de este shellscript.

Progresión aritmética de N términos.

```
#EJERCICIO 7
#!/bin/bash

# Verificar si el número de parámetros es correcto
if [ "$#" -ne 1 ]; then
  echo "Número de parámetros incorrecto"
  exit 1
fi

numero=1
maximo=$1

#Imprimo numero si este es <= maximo
while (test $numero -le $maximo)
do
      echo $numero

      #Sumo 1 a numero
      numero=`expr $numero + 1`
done;
```

```
Sergio@Sergio-PC MINGW64
$ ./ejercicio7_1.sh 8
1
2
3
4
5
6
7
8
```

Progresión geométrica de N términos.

```bash
#!/bin/bash

# Verificar si el número de parámetros es correcto
if [ "$#" -ne 1 ]; then
  echo "Número de parámetros incorrecto"
  exit 1
fi

numero=1
maximo=$1

#Imprimo numero si este es <= maximo
while (test $numero -le $maximo)
do
      echo $numero

      #Multiplico numero por 2
      numero=`expr $numero \* 2`
done;
```

```
Sergio@Sergio-PC MINGW64
$ ./ejercicio7_2.sh 18
1
2
4
8
16
```

Ejercicio 8

08	Escriba un shellscript que, utilizando el bucle "while", muestre el factorial de un número por pantalla. A continuación, haga lo mismo con un bucle "until". El número sobre el cual calcular el factorial será el único parámetro de este shellscript ($1). Recuerde que el factorial de un número es el producto de todos los números naturales desde 1 hasta ese número. Por ejemplo, el factorial de 6 (6!) será igual a 6 · 5 · 4 · 3 · 2 · 1.

Con while

```
#EJERCICIO 8
#!/bin/bash

# Verificar si el número de parámetros es correcto
if [ "$#" -ne 1 ]; then
  echo "Número de parámetros incorrecto"
  exit 1
fi

divisor=2
factorial=1
i=1

#Compruebo si i <= numero y si es >= 0
while (test $i -le $1 && test $i -ge 0)
do
      #Factorial es el valor de i * el valor de factorial
      factorial=`expr $i \* $factorial`

      #Sumo 1 a la variable i
      i=`expr $i + 1`
done
echo "El factorial de $1 es $factorial"
```

```
Sergio@Sergio-PC MINGW64
$ ./ejercicio8_1.sh 6
El factorial de 6 es 720
```

Con until

```
#!/bin/bash

# Verificar si el número de parámetros es correcto
if [ "$#" -ne 1 ]; then
  echo "Número de parámetros incorrecto"
  exit 1
fi

divisor=2
factorial=1
i=1

#Compruebo si i no es mayor que numero y si no es menor que 0
until [ $i -gt $1 -o $i -lt 0 ]
do
      #Factorial es el valor de i * el valor de factorial
      factorial=`expr $i \* $factorial`

      #Sumo 1 a la variable i
      i=`expr $i + 1`
done

# Mostrar el resultado al usuario
echo "El factorial de $1 es $factorial"
```

```
Sergio@Sergio-PC MINGW64 /h/IE
$ ./ejercicio8_2.sh 10
El factorial de 10 es 3628800
```

Ejercicio 9

09	Escriba un shellscript que intente autodestruirse (como archivo). Antes de ejecutar este shellscript, haga una copia de seguridad de su trabajo, por si acaso. No puede utilizar el nombre del archivo para eliminarlo. Comente el comportamiento del shellscript y si se borra o no el archivo.

```
#EJERCICIO 9
#!/bin/bash

#Copio el archivo ejercicio9.sh y le hago una copia
cp ./$0 ./$0.bak

echo "Se ha creado una copia de seguridad de su trabajo"

#Elimino el archivo archivo9.sh
rm $0;
```

```
Sergio@Sergio-PC MINGW64 /h/IES/1r DAW/Sistemes Inf
$ ./ejercicio9.sh
Se ha creado una copia de seguridad de su trabajo

Sergio@Sergio-PC MINGW64 /h/IES/1r DAW/Sistemes Inf
```

Ejercicio 10

10	Escriba un shellscript que, dados dos números diferentes pasados como parámetros, muestre por pantalla los números que van desde uno hasta el otro (ambos incluidos, crecientemente o decrecientemente). Debe resolver el ejercicio con un bucle until.

```
#EJERCICIO 10
#!/bin/bash

# Verificar si el número de parámetros es correcto
if [ "$#" -ne 2 ]; then
  echo "Número de parámetros incorrecto"
  exit 1
fi

numero1=$1
numero2=$2
menor=0
mayor=0

#Compruebo que número es menor
if (test $numero1 -lt $numero2)
then
      menor=$numero1
      mayor=$numero2
elif (test $numero1 -gt $numero2)
then
      menor=$numero2
      mayor=$numero1
else
      echo "Los números son iguales"
fi

#Imprimo desde el menor hasta el mayor
until (test $menor -gt $mayor)
do
      echo $menor
      menor=`expr $menor + 1`
done;
```

```
Sergio@Sergio-PC MINGW64
$ ./ejercicio10.sh 10 15
10
11
12
13
14
15
```

Ejercicio 11

11	Escriba un shellscript que, a partir del archivo /etc/passwd, diga cuál es el intérprete de comandos más utilizado por los usuarios del sistema (es decir, aquel que utilizan más usuarios). Este shellscript se puede resolver con una única (y larga) comando y sin utilizar ningún tipo de bucle.

```
#EJERCICIO 11
#!/bin/bash

interprete_mas_usado=$(cut -d: -f7 /etc/passwd | sort |
uniq -c | sort -rn | head -n1 | tr -s ' ' | cut -f2 | cut -d/ -f4)

echo "El intérprete de comandos más usado es \"$interprete_mas_usado\""
```

```
sergio@sergio-VirtualBox:~/Escritorio$ ./ejercicio11.sh
El intérprete de comandos más usado es "nologin"
```

Ejercicio 12

12	Elabore un shellscript que reciba como parámetros un número indeterminado de palabras en minúscula. Lo que tendrá que hacer el shellscript será agregar cada una de las palabras a archivos que se llamarán como su inicial.

```
#EJERCICIO 12
#!/bin/bash

#Recorre cada palabra
for palabra in "$@"
do
      #Obtiene la inicial de la palabra
      inicial=$(echo $palabra | head -c 1)

      #Agrega la palabra al archivo que corresponde
      echo $palabra >> ${inicial}.txt
done
echo "Ha funcionado"
```

```
Sergio@Sergio-PC MINGW64 /h/IES/1r DAW/Si
$ ./ejercicio12.sh hola pato gato muerto
Ha funcionado
```

Ejercicio 13

13	Indique justificadamente cuál es la función del siguiente shellscript, indicando cuál es el significado más lógico de los parámetros.

El primer parámetro recibe la carpeta que contiene los archivos para recorrerlos, el segundo es la palabra que queremos buscar. Finalmente, si encuentra la palabra imprime A y lo copiará en el directorio /tmp sino imprime B.