5. EJERCICIOS SCRIPTS

1. Crea un programa que reciba una ruta absoluta por parámetro y devuelva un mensaje indicando si es un fichero, un directorio o no existe en el sistema.

```
#!/bin/bash
if  [ $# -ne 1 ]
then
        echo "no has introducido ninguna ruta absoluta"
elif [ -e $1 ]
then
        if [ -d $1 ]
        then
                echo "$1 es un directorio"
        else
                echo "$1 es un fichero"
        fi
else
        echo "$1 No existe."
fi
```

2. Crea un programa que reciba un número indeterminado de parámetros y los imprima por pantalla con el siguiente formato:

El parámetro 1 es ….
El parámetro 2 es ….

```
#!/bin/bash
i=1
for k in $*
do
        echo El parámetro $i es $k
        #i=`expr $i + 1`
        let i=$i+1
done
```

3. Crea un programa que reciba como parámetro el nombre de un usuario del sistema e indique si está conectado o no.

```
#!/bin/bash
c=`who | grep -w $1 | tr -s ' ' | cut -d ' ' -f1 2>/dev/null`
if [ -z $c ]
then
        echo "$1 no está conectado"
else
        echo "$1 está conectado"
fi
```

4. Crea un guión shell que reciba como parámetro un número e indique si es primo o no.

```
#!/bin/bash
#k=`expr $1 - 1`
let k=$1-1
while [ $k -gt 1 ]
do
        #resto=`expr $1 % $k`
        let resto=$1%$k
        if [ $resto -eq 0 ]
        then
                echo El número $1 no es primo
                exit 0
        fi
        #k=`expr $k - 1`
        let k=$k-1
done
echo El número $1 es primo.
```

5. Crea un guión shell que reciba como parámetro la ruta competa de un directorio y muestre por pantalla la suma del tamaño de todos los ficheros que contiene.

```
#!/bin/bash

tot=0
for k in `ls -l $1 | tr -s ' ' ' ' | cut -d ' ' -f5`
do
        #tot=`expr $tot + $k`
        let tot=$tot+$k
done
echo El tamaño de los ficheros de $1 es $tot
```

6. Crea un programa que muestre un menú para realizar las siguientes operaciones:

   - Crear un grupo
   - Eliminar un grupo
   - Crear un usuario
   - Eliminar un usuario
   - Salir

   Se creará un bucle infinito, para que la única opción de salir sea que el usuario lo seleccione como opción.

   Haced el ejercicio con If y con CASE.

```
#!/bin/bash

while [ 1 -eq 1 ]
do
        echo 1 – crear grupo
        echo 2 – eliminar grupo
        echo 3 – crear usuario
        echo 4 – eliminar usuario
        echo 5 – salir
        echo introduzca una opcion
        read op
case $op in
1)
        echo introduzca el grupo
        read gr
        sudo addgroup $gr;;
2)
        echo introduzca el grupo
        read gr
        sudo delgroup $gr;;
3)
        echo introduzca el usuario
        read usu
        sudo useradd $usu;;
4)
        echo introduzca el usuario
        read usu
        sudo deluser $usu;;
5)
        exit 0;;
*)
        echo "no has introducido una opción válida";;
```

```
esac
done
```

****

```
#!/bin/bash

while [ 1 -eq 1 ]
do
        echo 1 – crear grupo
        echo 2 – eliminar grupo
        echo 3 – crear usuario
        echo 4 – eliminar usuario
        echo 5 – salir
        echo introduzca una opcion
        read op
if [ $op -eq 1 ]
then
        echo introduzca el grupo
        read gr
        sudo addgroup $gr
fi
if [ $op -eq 2 ]
then
        echo introduzca el grupo
        read gr
        sudo delgroup $gr
fi
if [ $op -eq 3 ]
then
        echo introduzca el usuario
        read usu
        sudo adduser $usu
fi
if [ $op -eq 4 ]
then
        echo introduzca el usuario
        read usu
        sudo deluser $usu
fi
if [ $op -eq 5 ]
then
exit 0
fi
done
```

7. Construye un programa que reciba como parámetros el nombre de un directorio y un tamaño en bytes. El programa debe mostrar un listado de todas las entradas de ese directorio, cuyo tamaño sea inferior al indicado por parámetro.
#!/bin/bash

```
for k in `ls -l $1 | tr -s ' ' '|'`
do
 nom=`echo $k | cut -d '|' -f9`
 tam=`echo $k | cut -d '|' -f5`

 if [ $tam -lt $2 ]
 then
 echo "$nom   $tam"
 fi
done
```

8. Crea el siguiente guión shell:

```
echo “Número $1. Total $*”
shift
echo “Número $1. Total $*”
shift
echo “Número $1. Total $*”
shift
```

Ejecútalo con los parámetros 1 2 3 4 5 y analiza los resultados.
Número 1. Total 1 2 3 4 5
Número 2. Total 2 3 4 5
Número 3. Total 3 4 5

Elimina el primero y desplaza la lista hacia la izquierda una posición.

9. Crea un programa que reciba por parámetro el nombre de un usuario y MATE a todos sus procesos.

```
#!/bin/bash

#for p in `ps -u $1 | tr -s ' ' ' ' | cut -d ' ' -f1`
#do
 #kill -9 $p
 #echo "he matado el proceso $p"
#done

#************************************
```

```
for p in `ps -u $1 | tr -s ' ' ' ' | cut -c2-  | cut -d ' ' -f 1`
do
 #kill -9 $p
 echo "he matado el proceso $p"
done
```

10. Crea un programa que mire cada 10 segundos los procesos existentes en memoria, y MATE a aquellos procesos cuyo PID esté en una lista almacenada en el fichero #!/bin/bash

```
#!/bin/bash
while [ 1 -eq 1 ]
do
 for p in `ps -ef | tr -s ' ' '|' | cut -d '|' -f2,8`
 #-ef para mostrar todos los procesos del sistema
 do
      pid=`echo $p | cut -d '|' -f1`
      nom=`echo $p | cut -d '|' -f2`
      b=`cat /root/procesos_capados | grep -w  $pid`

      if ! [ -z $b ]
      then
            #kill -9 $pid
            echo "he matado el proceso $nom con PID: $pid"
      fi
 done
sleep 10
done
```

11. Crea un programa capaz de gestionar el fichero del ejercicio 10. Las opciones que debe contener son:

    - ◦Listar procesos prohibidos.
    - ◦Añadir proceso.
    - ◦Eliminar proceso.

```
#!/bin/bash

while [ 1 -eq 1 ]
do
 echo 1 – Listar procesos
 echo 2 – Añadir proceso
 echo 3 – Eliminar proceso
 echo 4 – salir
 read op

 if [ $op -eq 1 ]
 then
        cat /root/procesos_capados
 fi

 if [ $op -eq 2 ]
 then
        read -p "¿qué PID de proceso quieres añadir?  " proceso
        echo $proceso >> /root/procesos_capados
        sort -t \n -k1n /root/procesos_capados
 fi

 if [ $op -eq 3 ]
 then
        read -p "¿qué PID de proceso quieres eliminar?  " proceso1

        for var in `cat /root/procesos_capados`
        do
                if [ $var != $proceso1 ]
                then
                        echo $var >> aux
                fi
        done
        mv aux /root/procesos_capados
        sort -t \n -k1n /root/procesos_capados
 fi

 if [ $op -eq 4 ]
 then
        exit
 fi
done
```