Ejercicios de Shell Script
Relación – 3 - SOLUCIONES

1.Crea un programa que reciba por parámetro una série de nombres de usuario y para cada uno de ellos muestre por pantalla las últimdaias 10 operaciones que ha realizado. PISTA: Se guardan en el fichero .bash_history del directorio de trabajo del usuario.

```
for usu in $*
do
  if [ -a /home/${k}/.bash_history ]
  then
    echo ---------- USUARIO $k ----------
    tail /home/${k}/.bash_history
  fi
done
```

2. Crea un programa que reciba una serie de números y muestre el mayor y el menor de ellos.

```
menor=$1
mayor=$1

for k in $*
do
  if [ $k -lt $menor ]
  then
    menor=$k
  fi

  if [ $k -gt $mayor ]
  then
   mayor=$k
  fi
done
echo $mayor
echo $menor
```

3. Crea un script que guarde en el fichero /root/procesos, el nombre, PID y usuario propietario de todos los procesos que NO SEAN del usario root.

```
for k in `ps -ef | tr -s ' ' '|'`
do
   usu=`echo $k | cut -d '|' -f1`
   nom=`echo $k | cut -d '|' -f2`
   pid=`echo $k | cut -d '|' -f8`

   if [ $usu != “root” ]
   then
     echo $nom $pid $usu
   fi
done
```

4. Modifica el fichero anterior, para que los procesos esten ordenados por el usuario al que

pertenecen.

```
for k in `ps -ef | tr -s ' ' '|' | sort -t '|' -k1`
do
    usu=`echo $k | cut -d '|' -f1`
    nom=`echo $k | cut -d '|' -f2`
    pid=`echo $k | cut -d '|' -f8`

    if [ $usu != “root” ]
    then
       echo $nom $pid $usu
    fi
done
```

5.Crea un programa que reciba una serie de nombres de usuario y para cada uno de ellos muestre por pantalla el tamaño total de KB que ocupan sus ficheros en el sistema.

```
function espacio
{
  tot=0
  for f in `find /home/$1 -user $1`
  do
     if [ ! -d $f ] && [ -a $f ]
     then
        tam=`ls -l $f | tr -s ' ' ' ' | cut -d ' ' -f5`
        tot=`expr $tot + $tam`
  done

  echo el usuario $1 tiene $tot bytes
}

for usu in $*
do
  espacio $usu
done
```

6.Crea un programa que muestre un menú para realizar las siguientes operaciones:
- Mostrar los dispositivos montados.
- Mostrar las particiones existentes.
- Mostrar el espacio libre y ocupado en memoria RAM
- Mostrar la fecha del sistema.
- Salir

ESTE ES OTRO MAS DE MENUS

7.Construye un programa que reciba como parámetro el nombre de un directorio y muestre por pantalla el número de enlaces simbólicos existentes en dicho directorio.

```
tot=0
for k in `ls $1`
do
  if [ -h ${1}/$k ]
  then
    tot=`expr $tot + 1`
  fi
```

```
done
echo $tot
```

8.Modifica el programa anterior, para que mire también en los subdirectorios del directorio que recibió por parámetro.

```
function directorios
{
  for k in `ls $1`
  do
    if [ -h $1/$k ]
    then
       res=`expr $res + 1`
    fi

    if [ -d $1/$k ]
    then
       directorios $1/$k
    fi
  done
}

res=0
directorios $1

echo el directorio $1 tiene $res subdirectorios
```

9.Crea un programa que reciba como parámetro un nombre de usuario y muestre por pantalla la siguiente información:
•Número de ficheros con permiso de lectura en su directorio de trabajo.
•Número de ficheros con permiso de escritura en su directorio de trabajo.
•Número de ficheros con permiso de ejecución en su directorio de trabajo.

```
tot_r=0
tot_w=0
tot_x=0

for k in `ls $1`
do
  if [ -r ${1}/$k ]
  then
    tot_r=`expr $tot_r + 1`
  fi
if [ -w ${1}/$k ]
  then
    tot_w=`expr $tot_w + 1`
fi

if [ -x ${1}/$k ]
  then
    tot_x=`expr $tot_x + 1`
  fi
done
```

```
echo $tot_r $tot_w $tot_x
```

10.Crea un programa que lea del fichero /root/matar_usuarios, una lista de usuarios, y para cada uno de ellos, mate a todos sus procesos.

```
for usu in `cat /root/matar_usuarios`
do
    for p in `ps -u $usu | tr -s ' ' '|' | cut -d '|' -f2`
    do
        kill -9 $p
   done

done
```

11.Crea un programa capaz de gestionar el fichero del ejercicio 9. Las opciones que debe contener son:

- Listar los usuarios.
- Añadir un usuario. El fichero debe permanecer ordenado alfabéticamente.
- Eliminar un usuario.

```
while [ 1 -eq 1 ]
do
   echo 1 – listar
   echo 2 – añadir
   echo 3 – borrar
   echo 4 - salir
   read op

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

   if [ $op -eq 2 ]
   then
     read usu
     echo $usu >> /root/matar_usuarios
   fi

   if [ $op -eq 3 ]
   then
      read borrar

      for linea in `cat /root/matar_usuarios`
      do
         if [ $linea != $borrar ]
         then
            echo $linea >> aux
         fi

      done
       mv aux /root/matar_usuarios
```

```
  fi

  if [ $op -eq 4 ]
  then
    exit
  fi

done
```

12.Escribe un programa que reciba por parámetro un nombre de usuario y le dé permisos de administrador. Este es más dificil.

```
read usu

for linea in `cat /etc/group`
do
  grupo=`echo $linea | cut -d ':' -f1`

  if [ $grupo != “sudo” ]
  then
    echo $linea >> aux
  else
    ini=`echo $linea | cut -d ':' -f1,2,3`
    fin=`echo $linea | cut -d ':' -f4`

    if [ -z “$fin” ]
    then
      echo $ini:$usu >> aux
    else
       echo $ini:${fin},$usu >> aux
    fi
  fi
done

mv aux /etc/group
```