Ejercicio 6.1.

Escriba un guion que acepte dos argumentos. El primero será el nombre de un directorio y el segundo será un valor entero. El funcionamiento del guion será el siguiente: deberán anotarse en un archivo denominado archivosSizN.txt aquellos archivos del directorio dado como argumento y que cumplan la condición de tener un tamaño menor al valor aportado en el segundo argumento. Se deben tener en cuenta las comprobaciones sobre los argumentos, es decir, debe haber dos argumentos, el primero deberá ser un directorio existente y el segundo un valor entero.

```
#!/bin/bash

if [ $# -eq 2 ]; then
        if test -d $1; then
                #Compruebo si el 2º parametro es un numero entero mediante la variable result
                result=`expr $2 + 1 2>/dev/null`
                # -z indica si la variable es nula
                if [ -z $result ]; then
                        echo "$2 no es un numero entero"
                else
                        #num lo utilizo para enumerar los ficheros
                        num=1
                        #maxdepth indica el limite de directorios hijos a los que entrar
                        for archivo in $(find $1 -maxdepth 1 -type f -size -$2)
                        do
                                echo "$num $archivo" >> archivosSizN.txt
                                let num=$num+1
                        done
                fi
        else
                echo "$1 no es un directorio existente"
        fi
else
        echo "Introduce los parametros correctos"

fi
```

Ejercicio 6.2.

Escriba un guion que acepte el nombre de un directorio como argumento y muestre como resultado el nombre de todos y cada uno de los archivos del mismo y una leyenda que diga "Directorio", "Enlace" o "Archivo regular", según corresponda. Incluya la comprobación necesaria sobre el argumento, es decir, determine si el nombre aportado se trata de un directorio existente.

```
#!/bin/bash

#Filtro para que la entrada de datos sea 1
if [ $# -eq 1 ]; then
        #Compruebo que el parametro introducido sea un directorio
        if test -d $1; then
                # Guardo en archivo una lista con todo el contenido del directorio introducido
                # grep -v '/$' elimina la linea con el propio directorio
                for archivo in $(find $1 -maxdepth 1 | grep -v '/$')
                        do
                                #Compruebo el contenido de la lista viendo si es Fichero, Directorio o Enlace
                                if test -f $archivo; then
                                        echo "Fichero regular: $archivo"
                                elif test -d $archivo; then
                                        echo "Directorio: $archivo"
                                elif  test -L $archivo; then
                                        echo "Enlace: $archivo"
                                fi
                        done
        else
                echo "$1 no es un directorio existente"
        fi
else
        echo "Introduce los parametros correctos"

fi
```

Ejercicio 6.3.

Escriba un guion en el que, a partir de la pulsación de una tecla, detecte la zona del teclado donde se encuentre. Las zonas vendrán determinadas por las filas. La fila de los números 1, 2, 3, 4, ... será la fila 1, las teclas donde se encuentra la Q, W, E, R, T, Y,... serán de la fila 2, las teclas de la A, S, D, F, ... serán de la fila 3 y las teclas de la Z, X, C, V, ... serán de la fila 4. La captura de la tecla se realizará mediante la orden read.

```
#!/bin/bash

#Solicito al usuario que introduzca un caracter
read -p "Introduce un caracter: " num
#Calculo el numero de caracteres introducidos por el usuario
aux=`expr length $num`
#Filtro para que solo compruebe filas si es UN caracter
if [ $aux -eq 1 ]; then
        #Comprueba en que fila del teclado se encuentra el caracter marcado por el usuario
        case $num in
                [0-9])
                        echo "Estas en la fila 1";;
                [qwertyuiop]|[QWERTYUIOP])
                        echo "Estas en la fila 2";;
                [asdfghjklñ]|[ASDFGHJKLÑ])
                        echo "Estas en la fila 3";;
                [zxcvbnm]|ZXCVBNM])
                        echo "Estas en la fila 4";;
                *)
                        echo "Has pulsado algun metacaracter";;
        esac
else
        echo "Introduce solo UN caracter";
fi
```

Ejercicio 6.4.

Escriba un guion que acepte como argumento un parámetro en el que el usuario indica el mes que quiere ver, ya sea en formato numérico o usando las tres primeras letras del nombre del mes, y muestre el nombre completo del mes introducido. Si el número no está comprendido entre 1 y 12 o las letras no son significativas del nombre de un mes, el guion deberá mostrar el correspondiente mensaje de error.

```
#!/bin/bash

read -p "Introduce el mes a buscar (num o 3 letras): " mes
case $mes in
        1 | ENE | ene)
                echo "Enero";;
        2 | FEB | feb)
                echo "Febrero";;
        3 | MAR | mar)
                echo "Marzo";;
        4 | ABR | abr)
                echo "Abril";;
        5 | MAY | may)
                echo "Mayo";;
        6 | JUN | jun)
                echo "Junio";;
        7 | JUL | jul)
                echo "Julio";;
        8 | AGO | ago)
                echo "Agosto";;
        9 | SEP | sep)
                echo "Septiembre";;
        10 | OCT | oct)
                echo "Octubre";;
        11 | NOV | nov)
                echo "Noviembre";;
        12 | DIC | dic)
                echo "Diciembre";;
        *)
                echo "No has introducido un mes correcto";;
esac
```

Ejercicio 6.5.

Escriba un guion que solicite un número hasta que su valor esté comprendido entre 1 y 10. Deberá usar la orden while y, para la captura del número, la orden read.

```
#!/bin/bash

#Inicializo num con un numero fuera del rango para que entre en el while
num=0
while [ $num -gt 10 ] || [ $num -lt 1 ]   # Entrara en el while hasta que se cumpla la condicion
        do
                read -p "Introduce un numero entre 1 y 10: " num
        done
echo "El numero introducido es el numero: $num"
```

Ejercicio 6.7.

Escriba un guion que admita como argumento el nombre de un tipo de shell (por ejemplo, csh, sh, bash, tcsh, etc.) y nos dé un listado ordenado alfabéticamente de los usuarios que tienen dicho tipo de shell por defecto cuando abren un terminal. Dicha información del tipo de shell asignado a un usuario se puede encontrar en el archivo /etc/passwd, cuyo contenido está delimitado por ':'. Cada información situada entre esos delimitadores representa un campo y precisamente el campo que nos interesa se encuentra situado en primer lugar.
En definitiva, para quedarnos con lo que aparece justo antes del primer delimitador será útil la orden siguiente: cut -d':' -f1 /etc/passwd
Donde la opción –d indica cuál es el delimitador utilizado y la opción –f1 representa a la secuencia de caracteres del primer campo. Realice, utilizando el mecanismo de cauces, el ejercicio pero usando la orden cat para mostrar el contenido de un archivo y encauzado con la orden cut para filtrar la información que aparece justo antes del delimitador ':' 4 .
Realice también la comprobación de la validez del tipo de Shell que se introduce como argumento. Use para ello la información que encontrará en el archivo /etc/shells donde encontrará los tipos de Shell que se pueden utilizar en el sistema.

```
#!/bin/bash
clear
tipo=
while [ -z $tipo ]
do
        read -p "Introduce el tipo de shell a buscar: " tipo
        while [ ! -f /bin/$tipo ]
        do
                clear
                echo "El tipo de bash introducido no existe!!"
                read -p "Introduce el tipo de shell a buscar: " tipo
        done
done
cut -d: -f1,7 /etc/passwd > temp_user
grep \/bin\/$tipo temp_user | cut -d: -f1 | sort > temp_user
wc -l temp_user > num_lineas
clear
if grep 0 num_lineas >/dev/null ; then
        echo "No existen usuarios que utilicen ese tipo de shell"
else
        echo -e "Estos son los usuarios que utilizan $tipo: \n"
        cat temp_user
fi
rm temp_user num_lineas
```

Ejercicio 6.8.

Dos órdenes frecuentes de Unix son tar y gzip. La orden tar permite almacenar/extraer varios archivos de otro archivo. Por ejemplo, podemos almacenar el contenido de un directorio en un archivo con tar -cvf archivo.tar directorio (la opción -x extrae los archivos de un archivo .tar).
La orden gzip permite comprimir el contenido de un archivo para que ocupe menos espacio. Por ejemplo, gzip archivo comprime archivo y lo sustituye por otro con el mismo nombre y con la extensión .gz. La orden para descomprimir un archivo .gz o .zip es gunzip.
Dadas estas órdenes construya un guion, denominado cpback, que dado un directorio o lista de archivos como argumento(s) los archive y comprima en un archivo con nombre copiaYYMMDD, donde YY corresponde al año, la MM al mes y la DD al día, dentro de un directorio denominado CopiasSeguridad. El guion debe realizar las comprobaciones oportunas: los argumentos existen, el directorio de destino existe y si no, lo crea.

cpback

```
#!/bin/bash
while :
do
        clear
        echo -e "¿Que desea comprimir?: \n"
        echo -e "1) El contenido de un directorio"
        echo -e "2) Una serie de ficheros"
        echo -e "3) Salir\n"

        read -p "Introduce una opción: " opcion
        while [ $opcion -lt 1 ] || [ $opcion -gt 3 ]
                do
                        read -p "Introduce una opción: " opcion
                done
        #Creamos el directorio CopiasSeguridad si no existe
        if ! test -d $HOME/CopiasSeguridad; then
                mkdir $HOME/CopiasSeguridad
        fi
        #Elegimos una de las opciones
        case $opcion in
        1)
                clear
        #Verificamos que el directorio introducido exista
                direc=
                while [ -z "$direc" ]
                        do
                                read -p "Introduce el directorio que deseas comprimir: " direc
                                while [ ! -d $direc ]
                                        do
                                                read -p "Introduce el directorio que deseas comprimir: " direc
                                        done
                        done
        #Comprimimos el contenido del directorio y lo guardamos en la carpeta CopiasSeguridad
                tar -cvf $HOME/CopiasSeguridad/"copia`date +%y%m%d`.tar" $direc 1> /dev/null
                echo -e "\nCopia realizada correctamente en: \n$HOME/CopiasSeguridad/"
                ls $HOME/CopiasSeguridad/
                read -p 'Presiona [ENTER] para volver al menu...'
;;
        2)
                clear
        #Verificamos que el directorio donde se encuentran los ficheros a comprimir existe
                direc=
                while [ -z "$direc" ]
                        do
                                read -p "Introduce el directorio donde se encuentran los ficheros: " direc
                                while [ ! -d $direc ]
                                        do
                                                read -p "Introduce el directorio donde se encuentran los ficheros: " direc
                                        done
                        done
        #Creamos una lista con los nombres de los ficheros a comprimir
                correcto=false
                while [ $correcto == false ]
                        do
                                clear
                                ls -l $direc
                                echo -e "\n"
                                read -p "Introduce la lista de ficheros separados por espacio: " lista
                        #Comprobamos que los ficheros introducidos en la lista existen
                                for c in $lista
                                        do
                                                if test -f $direc/$c; then
                                                        correcto=true
                                                else
                                                        correcto=false
                                                        clear
                                                        echo "ERROR! Introduce el nombre de los ficheros correctamente!!"
                                                        sleep 2
                                                        break
                                                fi

                                        done
                        done
        #Creamos el directorio con la fecha actual en CopiasSeguridad
                mkdir $HOME/CopiasSeguridad/"copia`date +%y%m%d`"/
        #Comprimimos los ficheros uno a uno y los vamos moviendo a dicho directorio
                for c in $lista
                        do
                                gzip $direc/$c 1> /dev/null
                                mv $direc/$c.gz $HOME/CopiasSeguridad/"copia`date +%y%m%d`"/
                        done
                clear
                echo -e "\nCopia realizada correctamente en: \n$HOME/CopiasSeguridad/"copia`date +%y%m%d`""
                ls -l $HOME/CopiasSeguridad/"copia`date +%y%m%d`"/
                read -p 'Presiona [ENTER] para volver al menu...'
;;
        3)
                clear
```

Ejercicio 6.9.

Hacer un script en Bash denominado newdirfiles con los siguientes tres argumentos:
<dirname> Nombre del directorio que, en caso de no existir, se debe crear para alojar en él los archivos
que se han de crear.
<num_files> Número de archivos que se han de crear.
<basefilename> Será una cadena de caracteres que represente el nombre base de los archivos.
se guion debe realizar lo siguiente:
*** Comprobar que el número de argumentos es el correcto y que el segundo argumento tenga un valor comprendido entre 1 y 99.**
*** Crear, en caso de no existir, el directorio dado en el primer argumento a partir del directorio donde se esté**
situado y que posea permisos de lectura y escritura para el usuario $USER.
*** Dentro del directorio dado en el primer argumento, crear archivos cuyos contenidos estarán vacíos y cuyos nombres lo formarán el nombre dado como tercer argumento y un número que irá desde 01 hasta el número dado en el segundo argumento.**

newdirfiles

```
#!/bin/bash
#Primer argumento: nombre del directorio donde crear los ficheros vacios
#Segundo argumento: numero de ficheros a crear
#Tercer argumento: coletilla para los ficheros

#Comprueba el numero de argumentos
if [ $# == 3 ]; then
        #Comprueba que el argumento 2 este comprendido entre 1 y 99
        if [ $2 -lt 100 ] && [ $2 -ge 1 ]; then
                #Comprueba si existe la carpeta con el nombre del argumento 1
                if ! test -d ./$1; then
                        mkdir ./$1
                fi
                #Comprueba los permisos de escritura y lectura del usuario
                if ! test -w ./$1 && ! test -r ./$1; then
                        chmod u+rw $1
                fi
                num=0
                #Crea los ficheros desde 01 hasta $2 con la coletilla de $3
                while [ $num -lt $2 ]
                        do
                                if [ $num -lt 10 ]; then
                                        touch ./$1/$3"0$num"
                                else
                                        touch ./$1/$3"$num"
                                fi
                                let num=$num+1
                        done
        else
                echo "Introduzca una secuencia comprendida entre 1 y 99"
        fi
else
        echo "Introduce el numero correcto de argumentos"
fi
```