

**Practica 1**  
**PROGRAMACION SHELL**  
**INF – 113**

**PROBLEMAS RESUELTOS:**

1. Dado un nombre de directorio, un nombre de archivos y una cadena leída por teclado, verificar si el directorio leído existe en el directorio de trabajo, verificar si el directorio contiene el archivo leído por teclado y además si el archivos contiene la cadena leída por teclado, si cumpliese estas condiciones ordenar el contenido del archivo y guardarlo en un sub directorio llamado “ordenado” (crear el directorio si no existiera).

**ENTRADA:**

**Nombre de directorio**

registros

**Nombre de archivo**

Listado.txt

**Cadena a buscar**

Juan

**SALIDA:**

Archivo ordenado...

Directorio de Trabajo: /home/gonzalo/Documentos/script

ORIGEN: /home/gonzalo/Documentos/script/registros/Listado.txt

DESTINO: /home/gonzalo/Documentos/script/registros/ordenado/Listado.txt

**SOLUCIÓN:**

```
#!/bin/bash
echo Nombre de directorio
read d # d es el nombre de directorio a buscar
DIR=$(pwd)/$d # con pwd obtenemos el directorio actual
if [ -d $DIR ];then # verificando si el directorio existe
    echo Nombre de archivo
    read f # f nombre de archivo
    FILE=$DIR/$f
    if [ -f $FILE ];then # verificando si el directorio contiene el archivo 'f'
        echo Cadena a buscar
        read c # ingresando cadena a buscar
        if grep -i $c $FILE ; then # buscando si existe la cadena en el archivo
            clear
            if [ ! -d $DIR/ordenado ];then
                mkdir $DIR/ordenado # creando subdirectorio si no existiera
            fi
            sort $FILE > $DIR/ordenado/$f # guardando el archivo ya ordenado
            echo Archivo ordenado... # en el directorio ordenado
            echo Directorio de Trabajo: $(pwd)
            echo ORIGEN: $FILE
            echo DESTINO: $DIR/ordenado/$f
        else
            echo No existe contenido;
        fi
    else
        echo No existe el archivo '$FILE'
    fi
else
    echo No existe el directorio '$DIR'
fi
```

```
Archivo ordenado...
Directorio de Trabajo: /home/gonzalo/Documentos/script
ORIGEN: /home/gonzalo/Documentos/script/registros/Listado.txt
DESTINO: /home/gonzalo/Documentos/script/registros/ordenado/Listado.txt
```

Archivo original:

```
gonzalo@gonzalo-RV408-RV508:~/Documentos/script$ head -v registros/Listado.txt
==> registros/Listado.txt <==
Gonzalo Mamani G.
Juan Lechin O.
Alverto Lopez Obregon.
```

Archivo ordenado:

```
gonzalo@gonzalo-RV408-RV508:~/Documentos/script$ head -v registros/ordenado/Listado.txt
==> registros/ordenado/Listado.txt <==
Alverto Lopez Obregon.
Gonzalo Mamani G.
Juan Lechin O.
```

2. Dado un archivo (lista.txt) que contiene la lista completa de los empleados en una empresa, se requiere clasificar por el primer nombre de los empleados en diferentes archivos, dichos archivos clasificados se deben almacenar en un sub directorio llamado “clasificados” (crear directorio si no existiera):

Obs. El archivo con los datos posee la siguiente estructura:

lista.txt

Nombre Completo	Apellido Paterno	Apellido Materno

### ENTRADA:

### SALIDA:

Archivo Clasificado...

### SOLUCIÓN:

```
#!/bin/bash
ARCH=$(pwd)/lista.txt # obteniendo la direccion del archivo
if [ -f $ARCH ];then # verificando se existe el archivo
    DIR=$(pwd)/clasificado # obteniendo la direccion del directorio destino
    while read line; do # recorremos linea por linea el archivo lista.txt
        for c in $line;do # recorre palabra por palabra
            if [ ! -d $DIR ]; then # verificamos si no existe el directorio destino
                mkdir clasificado # creamos directorio
            fi
            echo $line >> $DIR/$c.txt # adicionamos en el archivo que perteneces
            break; # salimos del ciclo despues de una iteracion
        done
    done < $ARCH # archivo a evaluar linea por linea
    echo Archivo Clasificado...
else
    echo No existe Archivo...
fi
```

Archivo Origen (lista.txt):

```
gonzalo@gonzalo-RV408-RV508:~/Documentos/script$ head -v lista.txt
==> lista.txt <==
Juan Diaz Miranda
Pedro Lopez Arena
Ivan Chipana Herrero
Juan Moreno Altamirano
Juan Alverto Zelada Lopez
Pedro Moredno Llanque
Pedro Alverto Miranda Moreno
Ivan Villanueva Sanchez
```

Resultado 3 archivos: (clasificado/Ivan.txt, clasificado/Juan.txt, clasificado/Pedro.txt):

```
gonzalo@gonzalo-RV408-RV508:~/Documentos/script$ head -v clasificado/Ivan.txt
==> clasificado/Ivan.txt <==
Ivan Chipana Herrero
Ivan Villanueva Sanchez
gonzalo@gonzalo-RV408-RV508:~/Documentos/script$ head -v clasificado/Juan.txt
==> clasificado/Juan.txt <==
Juan Diaz Miranda
Juan Moreno Altamirano
Juan Alverto Zelada Lopez
gonzalo@gonzalo-RV408-RV508:~/Documentos/script$ head -v clasificado/Pedro.txt
==> clasificado/Pedro.txt <==
Pedro Lopez Arena
Pedro Moredno Llanque
Pedro Alverto Miranda Moreno
```

3. Dado un numero n ( $1 < n \leq 100$ ) verificar si el número es primo o no

<p><b>ENTRADA:</b> 5</p> <p><b>SALIDA</b> Es Primo</p>	<p><b>ENTRADA:</b> 150</p> <p><b>SALIDA</b> numero no valido</p>
--	--

**Solución:**

```
#!/bin/bash
echo Ingrese Numero
read n
if [ $n -le 100 -a $n -gt 1 ];then # 1<n<=100
    cont=2
    sw=0
    while [ $cont -le $n ];do # mientras cont<n
        num=`expr $n % $cont`
        if [ $num -eq 0 ];then
            sw=`expr $sw + 1`
        fi
        cont=`expr $cont + 1`
    done
    if [ $sw -eq 1 ];then
        echo "Es Primo"
    else
        echo "No es Primo"
    fi
else
    echo numero no valido
fi
```

```
Ingrese Numero
5
Es Primo
```

4. Dados dos números f (filas) y c (columnas) generar una estructura con las siguientes características:

**Ejemplo:**

si f=4 y c=4				si f=2 y c=3		
0	2	1	3	0	2	1
1	5	2	7	3	1	5
3	11	5	13			
8	17	13	19			

Análisis:

Para la solución del problema se debe analizar los elementos de la estructura, y tratarlos como una serie de número, podemos verificar lo siguiente:

0	2	1	3
1	5	2	7
3	11	5	13
8	17	13	19

Podemos concluir que se trata de una serie Fibonacci alternando con una serie de números primos.

**0 1 1 2 3 5 8 13** y **2 3 5 7 11 13 17 19**

### SOLUCIÓN:

```
#!/bin/bash
echo "Filas: "; read f
echo "Columnas: "; read c;
x=-1;y=1; #variables para el control de la serie fibonacci
sw=0; # control para alternar entre fibonacci y primo
p=1; # variable de la serie de primos
for i in $(seq 1 $f); do # equivale: i=1;i<=f;i++
    for j in $(seq 1 $c); do # equivale: j=1;j<=c;j++

        if [ $sw -eq 0 ];then # si sw es 0 generamos el siguiente fibonacci
            z=`expr $x + $y`
            echo -n " $z " # mostramos el numero fibonacci
            x=$y;y=$z
            sw=1
        else # si sw es 1 generamos el siguiente numero primo
            while [ true ];do # buscando el siguiente numero primo
                cont=2;cw=0; # inicializamos variables
                p=`expr $p + 1` # verificamos el siguiente numero
                while [ $cont -le $p ];do
                    num=`expr $p % $cont`
                    if [ $num -eq 0 ];then
                        cw=`expr $cw + 1`
                    fi
                    cont=`expr $cont + 1`
                done
                if [ $cw -eq 1 ];then # si cw es 1 entonces
                    break # el numero generado es primo
                fi
            done
            echo -n " $p " # mostramos el numero primero
            sw=0
        fi
    done
    echo ""
done
```

```
Filas:
4
Columnas:
4
0 2 1 3
1 5 2 7
3 11 5 13
8 17 13 19
```

5. Dada una cadena de caracteres en minúscula leída por teclado, mostrar el número de vocales y consonantes de cada palabra.

**ENTRADA:**

Hola mundo

**SALIDA:**

Palabra: 'hola' Nro. de vocales: 2 Nro. de consonantes: 2

Palabra: 'mundo' Nro. de vocales: 2 Nro. de consonantes: 3

**SOLUCIÓN:**

```
#!/bin/bash
echo Ingrese cadena:
read p
for i in $p; do
n=`expr length $i` #hallando longitud de la i-esima palabra
posicion=1
consonante=0
vocal=0
while [ $posicion -le $n ];do # equivale: while(posicion<=n)
car=$(echo $i | cut -c $posicion); # obtenemos el caracter en la posicion actual
if [ "$car" = "a" -o "$car" = "e" -o "$car" = "i" -o "$car" = "o" -o "$car" = "u" ]; then
vocal=`expr $vocal + 1`;
else
consonante=`expr $consonante + 1`;
fi
posicion=`expr $posicion + 1`;
done
echo "Palabra: '$i' Nro. de vocales: $vocal Nro. de consonantes: $consonante"
done
```

```
Ingrese cadena:
hola mundo
Palabra: 'hola' Nro. de vocales: 2 Nro. de consonantes: 2
Palabra: 'mundo' Nro. de vocales: 2 Nro. de consonantes: 3
```

**PROBLEMAS PROPUESTOS:**

1. Dados tres archivos (lista1.txt, lista2.txt y lista3.txt) de lista de empleados, que cumplan con la estructura de nombres, apellido paterno y apellido materno (Ver Problemas Resueltas, Preg. 2), leer una cadena que represente a un nombre (solo un nombre si tuviera dos nombre), y buscar el nombre en los tres archivos, si el nombre existe en los tres archivos entonces adicionar el nombre completo y los apellidos a un nuevo archivo llamado encontrados.txt.

**Ejemplo:**

**lista1.txt**

Juan Alverto	Rivera	Lopez
Maria	Miranda	Duran

**lista2.txt**

Alveto	Ramirez	Nina
Maria	Miranda	Duran
Juan	Apaza	Romero

**lista3.txt**

Pedro	Quispe	Lopez
Reinaldo	Diaz	Moreno
Juana Rosa	Apaza	Romero
Juan	Rivas	Landaeta

**ENTRADA:**

juan

**SALIDA:**

Empleado encontrado...

**encontrados.txt**

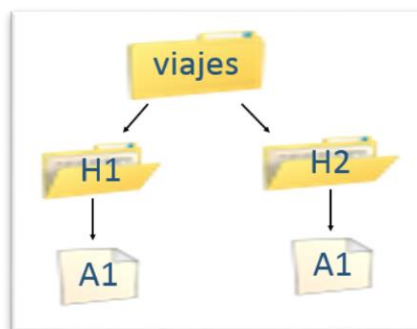
Juan Alveto	Rivera	Lopez
Juan	Apaza	Romero
Juan	Rivas	Landaeta

2. Generación de directorios y archivos dinámicos: dados tres parámetros numéricos mayores que cero leídos por teclado (P, H, A), generar directorios dinámicos en el directorio de trabajo.

**Ejemplos:**

Si P=1 y H=2 y A=1

Y el nombre para el directorio superior es “viajes”

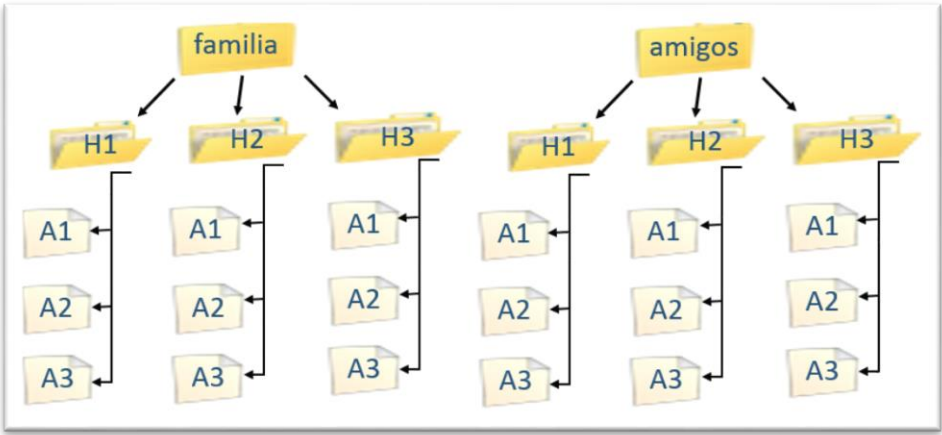


Esto nos dice que la carpeta “viajes” contendrá dos directorios H1 y H2 y cada sub directorio contendrá un archivo A1.

**Ejemplos 2:**

Si  $P=2$ ,  $H=3$  y  $A=3$

Y los directorios superiores serán “familia”, “amigos”



Esto nos dice que generaremos 2 carpetas “familia” y “amigos”, contendrán tres directorios H1, H2 y H3 y cada sub directorio contendrá tres archivos archivo A1, A2 y A3.

Obs. : Los nombres de los directorios superiores son leídos por teclado.

3. Dados dos números f (filas) y c (columnas) generar una estructura con las siguientes características:

**Ejemplo:**

si $f=6$ y $c=2$		si $f=5$ y $c=4$			
0	1	0	1	2	3
1	2	1	2	3	4
1	2	1	2	3	4
2	3	2	3	4	5
3	4	3	4	5	6
5	6				



4. Dados dos número: a ( $2 \leq a \leq 100$ ) y b ( $3 \leq b \leq 200$ ), si a es mayor que b mostrar la resta de a menos b, si a es igual a b mostrar la suma de a y b, y si a es menor a b mostrar la multiplicación de a y b;

**ENTRADA:**

Ingrese a:

20

Ingrese b:

5

**SALIDA:**

15

5. Dada una cadena de caracteres en minúscula leída por teclado, mostrar solo las palabras palíndromos.

**ENTRADA:**

mi familia y yo somos de oruro

**SALIDA:**

somos

oruro