



## Ejercicio 1:

El pedido test dispone de un largo número de funciones que permiten hacer comparaciones. Rellene las tres tablas siguientes, respectivas a los tres tipos de operadores de comparación de test. Indica qué valor tomaría la variable?, En función de los valores introducidos (recuerde que 0 es cierto a Linux).

<u>Comparación numérica</u>	
test n1 -eq n2	? valdría 0 si n1 és 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

<u>Comparación de cadenas de texto</u>	
test s1 = s2	? valdría 0 si s1 es igual a s2
test s1 != s2	? valdría 0 si s1 no es igual a s2
test -n s1	? valdría 0 si la longitud de s1 no es 0
test -z s1	? valdría 0 si la longitud de s1 es 0

<u>Comparación de ficheros</u>	
test -d f1	? valdría 0 si el f1 existe y es un directorio
test -e f1	? valdría 0 si el f1 existe
test -f f1	? valdría 0 si el f1 existe y es un fichero regular
test -r f1	? valdría 0 si el f1 existe y tiene permisos de lectura
test -s f1	? valdría 0 si el f1 existe y no está vacío
test -w f1	? valdría 0 si el f1 tiene permiso de escritura
test -x f1	? valdría 0 si el f1 tiene permiso de ejecución
test f1 -nt f2	? valdría 0 si el f1 es modificado/creado después que f2 (compruebe si f1 es más nuevo que f2)
test f1 -ot f2	? valdría 0 si el f1 es modificado/creado antes que f2 (comprobar si f1 es anterior a f2)



## ***Ejercicio 2:***

Explica el funcionamiento de las comillas de este ejercicio, y el por qué de ello.

```
#!/bin/bash

comanda=ls

echo "$comanda"
echo ` $comanda `
echo '$comanda'
```

- **echo "\$comanda"** = Muestra el valor de la variable "comanda".
- **echo ` \$comanda `** = Ejecuta el valor de la variable, que en este caso es el "ls".
- **echo '\$comanda'** = No muestra el valor, sino muestra por pantalla lo que esta en comillas simples.

## ***Ejercicio 3:***

Indicar justificadamente cuál es la función del siguiente shellsript, indicando cuál es el significado más lógico de los parámetros.

```
#!/bin/bash

touch tmp

for i in *.txt
do
    grep "examen" $i >> tmp
done

wc -l < tmp
rm tmp
```

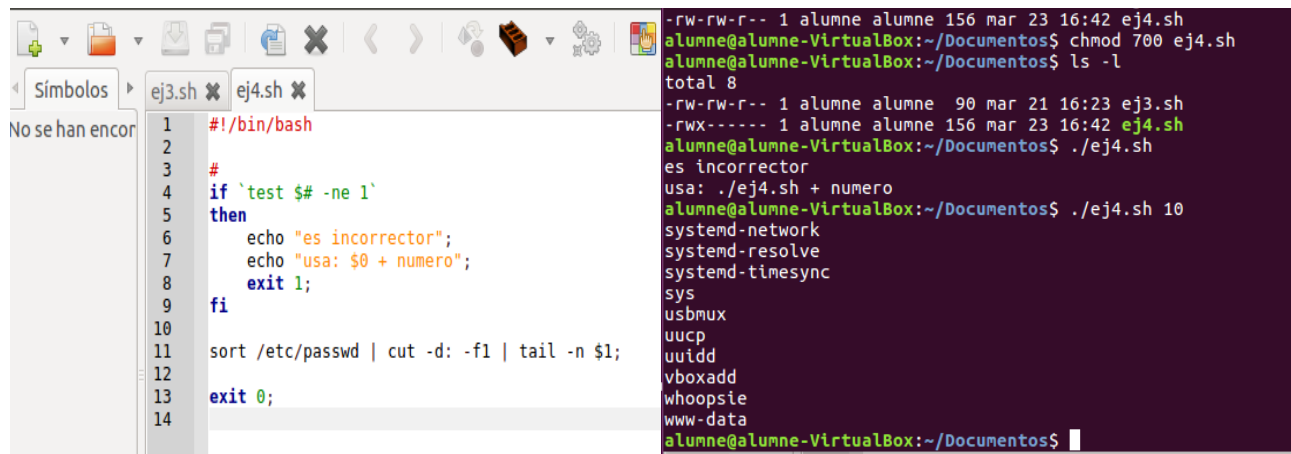
- **touch tmp** : Primero crea un fichero vacio llamado "tmp".
- **for i in \*.txt; do grep "examen" \$i >> tmp; done** : Después se crea el variable "i" y el valor los asigna cojiendo todos los ficheros txt y busca la palabra "examen" y este valor del variable "i" lo guarda en el fichero tmp creado anteriormente. Y acaba el proceso.
- Después el comando "**wc -l < tmp**" muestra el numero de las lineas del fichero "tmp".
- **rm tmp** : Y al final borra el fichero tmp.

***Escribir un comando equivalente a todo este shellsript.***

- **touch tmp; for i in \*.txt; do grep "examen" \$i >> tmp; done; wc -l tmp**

### Ejercicio 4:

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

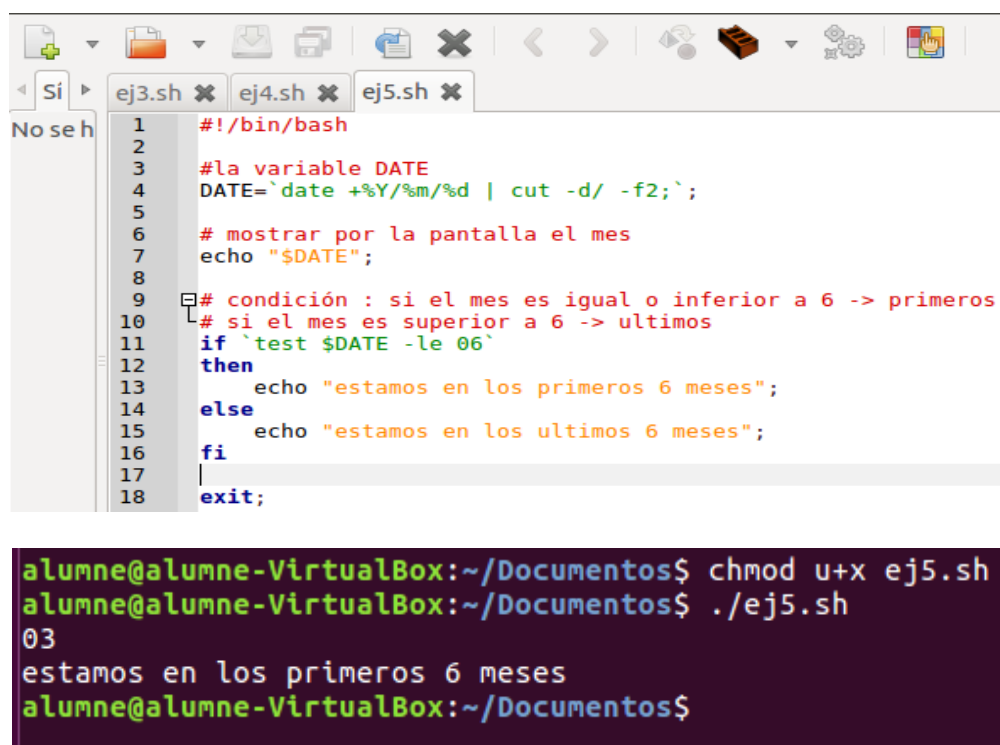


```
1 #!/bin/bash
2
3 #
4 if `test $# -ne 1`
5 then
6     echo "es incorrector";
7     echo "usa: $0 + numero";
8     exit 1;
9 fi
10
11 sort /etc/passwd | cut -d: -f1 | tail -n $1;
12
13 exit 0;
```

```
alumno@alumno-VirtualBox:~/Documentos$ chmod 700 ej4.sh
alumno@alumno-VirtualBox:~/Documentos$ ls -l
total 8
-rw-rw-r-- 1 alumno alumno 90 mar 21 16:23 ej3.sh
-rwx----- 1 alumno alumno 156 mar 23 16:42 ej4.sh
alumno@alumno-VirtualBox:~/Documentos$ ./ej4.sh
es incorrector
usa: ./ej4.sh + numero
alumno@alumno-VirtualBox:~/Documentos$ ./ej4.sh 10
systemd-network
systemd-resolve
systemd-timesync
sys
usbmux
uucp
uuidd
vboxadd
whoopsie
www-data
alumno@alumno-VirtualBox:~/Documentos$
```

### Ejercicio 5:

Escribir shellscript que indique si nos encontramos en primeros o los últimos seis meses del año. Hay que tener 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.



```
1 #!/bin/bash
2
3 #la variable DATE
4 DATE=`date +%Y/%m/%d | cut -d/ -f2`;
5
6 # mostrar por la pantalla el mes
7 echo "$DATE";
8
9 # condición : si el mes es igual o inferior a 6 -> primeros
10 # si el mes es superior a 6 -> ultimos
11 if `test $DATE -le 06`
12 then
13     echo "estamos en los primeros 6 meses";
14 else
15     echo "estamos en los ultimos 6 meses";
16 fi
17
18 exit;
```

```
alumno@alumno-VirtualBox:~/Documentos$ chmod u+x ej5.sh
alumno@alumno-VirtualBox:~/Documentos$ ./ej5.sh
03
estamos en los primeros 6 meses
alumno@alumno-VirtualBox:~/Documentos$
```

## Ejercicio 6:

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

```
ej3.sh ✕ ej4.sh ✕ ej5.sh ✕ ej6.sh ✕
1  #!/bin/bash
2
3  #
4
5  if `test $# -eq 0`
6  then
7
8      echo "Incorrecto, Usa $0 + parametros"
9      exit 1;
10 fi
11
12 # Poner palabra Hello delante cualquier parametro
13 for h in $*
14 do
15     echo "Hello $h";
16 done;
17
18 exit;
```

```
alumno@alumno-VirtualBox:~/Documentos$ chmod u+x ej6.sh
alumno@alumno-VirtualBox:~/Documentos$ ./ej6.sh
Incorrecto, Usa ./ej6.sh + parametros
alumno@alumno-VirtualBox:~/Documentos$ ./ej6.sh harman
Hello harman
alumno@alumno-VirtualBox:~/Documentos$
```

## Ejercicio 7:

Escribir 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 \$1 de este shellscript.

```
p7.sh - shells - Visual Studio Code
ej8.sh p7.sh ✕
1  #!/bin/bash
2  clear
3  n=1
4  echo "Ingresa un número y presiona ENTER"
5  read m
6  #El ciclo controla que n sea menor o igual a 10
7  while [ $n -le 10 ]
8  do
9
10     #En R almacenamos la multiplicación de n por m
11     r=$((n*m))
12     #Se imprime dicha multiplicación en pantalla
13     echo $r
14     #Con let, incrementamos el valor de n en 1 unidad
15     let n=n+1
16 done
17 exit;
```

```
alumno@alumno-VirtualBox: ~/Documentos/shells
"Ingresa un número y presiona ENTER"
2
2
4
6
8
10
12
14
16
18
20
alumno@alumno-VirtualBox:~/Documentos/shells$
```

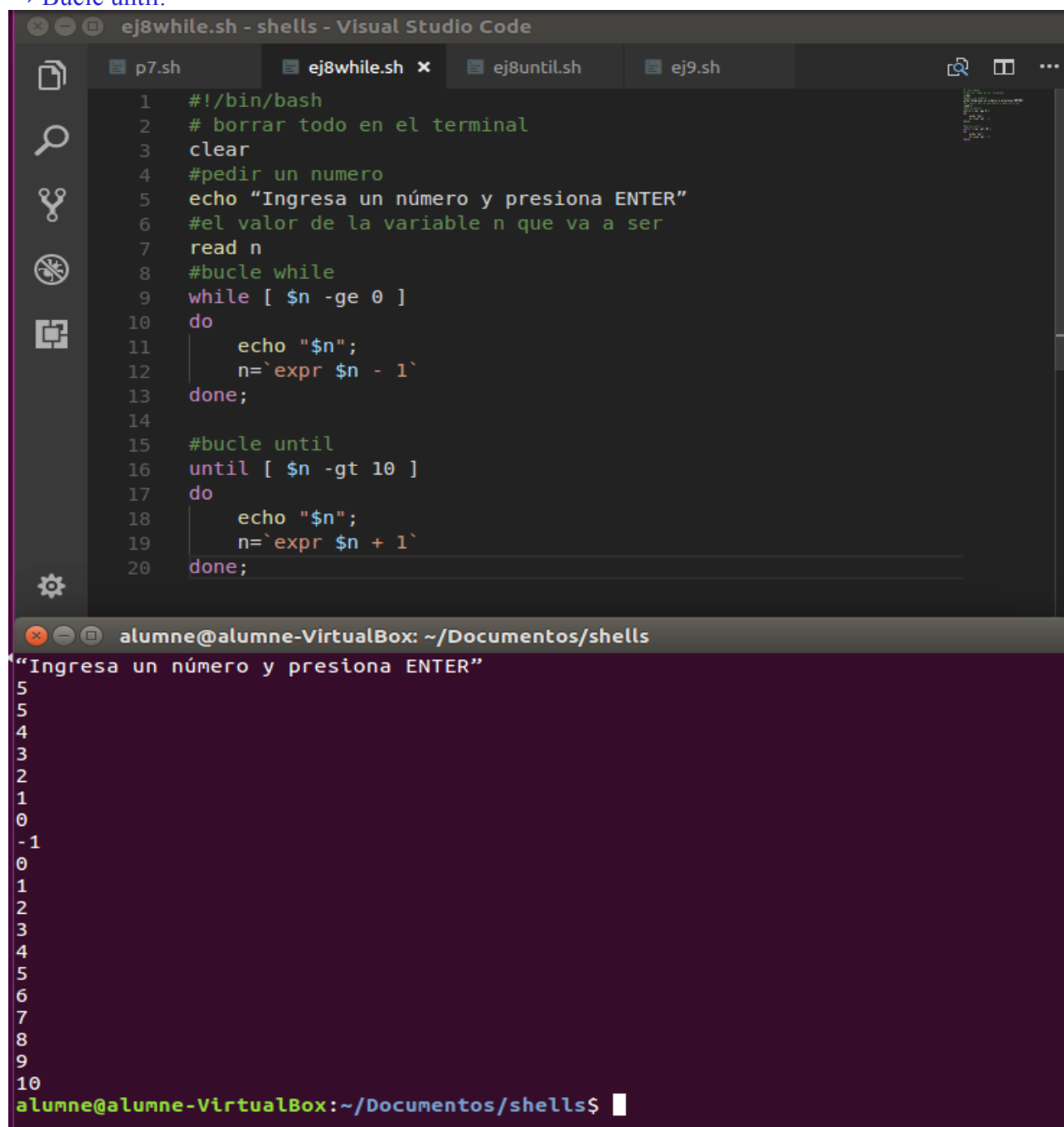
## Ejercicio 8:

Escribir 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 que 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 este número. Por ejemplo, el factorial de 6 (6!), Será igual a  $6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$ .*

→ Bucle while:

→ Bucle until:



```
ej8while.sh - shells - Visual Studio Code
1  #!/bin/bash
2  # borrar todo en el terminal
3  clear
4  #pedir un numero
5  echo "Ingresa un número y presiona ENTER"
6  #el valor de la variable n que va a ser
7  read n
8  #bucle while
9  while [ $n -ge 0 ]
10 do
11     echo "$n";
12     n=`expr $n - 1`
13 done;
14
15 #bucle until
16 until [ $n -gt 10 ]
17 do
18     echo "$n";
19     n=`expr $n + 1`
20 done;

alumne@alumne-VirtualBox: ~/Documentos/shells
"Ingresa un número y presiona ENTER"
5
5
4
3
2
1
0
-1
0
1
2
3
4
5
6
7
8
9
10
alumne@alumne-VirtualBox:~/Documentos/shells$
```