

# Estructuras de control y bucles

## comparaciones de cadenas alfanuméricas (Test Operators, Binary Comparison)

- cadena1 = cadena2 (también cadena1 == cadena2)
- cadena1 != cadena2
- cadena1 < cadena2
- cadena1 > cadena2
- -n cadena1 → longitud mayor que 0
- -z cadena1 → longitud = 0
- cadena =~ REG\_EXP
  - <http://stackoverflow.com/questions/2348379/use-regular-expression-in-if-condition-in-bash>
  - [http://wiki.bash-hackers.org/syntax/ccmd/conditional\\_expression](http://wiki.bash-hackers.org/syntax/ccmd/conditional_expression)

## comparaciones de valores numéricos (Test Operators, Binary Comparison)

- x -lt y → x < y
- x -le y → x ≤ y
- x -eq y → x = y
- x -ge y → x ≥ y
- x -gt y → x > y
- x -ne y → x ≠ y

si se pone entre doble paréntesis:

- x > y
- x >= y
- x < y
- x <= y

## comprobación atributos de fichero (Test Operators, Files)

- -d fichero → fichero existe y es un directorio
- -e fichero → fichero existe
- -f fichero → fichero existe y es un fichero regular (no un directorio o fichero especial)
- -s fichero → fichero existe y no está vacío
- -h fichero → fichero existe y es un symlink
  - -L?
- -b fichero → fichero existe y es un device block
- -c fichero → fichero existe y es un device character
- -p fichero → fichero existe y es un pipe
- -S fichero → fichero existe y es un socket
- -t fichero → fichero existe y está asociado a un terminal
- -N fichero → fichero existe y ha sido modificado desde su última lectura
- -r fichero → tiene permiso de lectura
- -w fichero → tiene permiso de escritura
- -x fichero → tiene permiso de ejecución o de búsqueda si es un directorio

- **-g** fichero → tiene el SGID
- **-u** fichero → tiene el SUID
- **-k** fichero → tiene el «sticky bit»
- **-O** fichero → eres el OWNER del fichero
- **-G** fichero → el GRUPO del fichero es el mismo que el tuyo
- fichero1 **-nt** fichero2 → el fichero1 es más reciente que el fichero2
- fichero1 **-ot** fichero2 → el fichero1 es más antiguo que el fichero2
- fichero1 **-ef** fichero2 → el fichero1 y el fichero2 son HARD LINKS al mismo fichero
- stat <fichero> → información en disco del fichero
  - con -c %s → tamaño del archivo

## concatenación de comparaciones / comprobaciones

- && → and
- || → OR
- ! → NOT

/vía: <http://www.linux-es.org/node/238>

## estructuras

- if/else

```
if condicion
then
    comandos
elif condicion
    comandos
else
    comandos
fi
DIRECTORIO="/tmp/test"
COMANDO="/bin/mkdir $DIRECTORIO"

if $COMANDO then
    echo "OK"
else
    echo "KO"
fi
```

- if (ternary operator)
  - a=\$( [ «\$b» == 5 ] && echo «\$c» || echo «\$d»)
  - 1)
  - [ \$b == 5 ] && { a=\$c; true; } || a=\$d
  - VARIABLE=`[ test ] && echo VALUE\_A || echo VALUE\_B`
- for

```
for nombre [in lista]
do
    comandos
done
```

```
LISTA="etc var home"
for directorio in $LISTA
do
    ls $directorio
done
```

- while

```
while condición
do
    comandos
done

NUM=0
while [ $NUM -gt 10]; do
    let NUM=$NUM+1
done
```

- while \$ while true; do sleep 10; ls -la; done
- until

```
until condición
do
    comandos
done

NUM=0
until [ $NUM -gt 10]; do
    let NUM=$NUM+1
done
```

- case

```
case expresion in
    case1)
        comandos
    case2)
        comandos
    ...
esac

for NUM in 0 1 2 3
do
    case $NUM in
        0) echo "cero";
        1) echo "uno";
        2) echo "dos";
        3) echo "tres";
    esac
done
```

- select
- break (y nested break) :
<https://unix.stackexchange.com/questions/200381/how-can-i-break-out-of-a-while-loop-from-within-a-nested-case-statement>

<sup>1)</sup>

a = b==5 ? c : d

From:  
<https://miguelangel.torresegea.es/wiki/> - **miguel angel torres egea**



Permanent link:  
<https://miguelangel.torresegea.es/wiki/linux:bash:estructurasdecontrol?rev=1530228827>

Last update: **28/06/2018 16:33**