ORDENES BASH

cd directorio
{Te lleva al directorio que indiques}

---

ls [opción] [archivo]
{Lista los archivos del directorio actual o del especificado}

-a	lista los archivos del directorio, incluidos aquellos cuyo nombre comienza con “.”.
-l	formato largo.
-r	lista en orden inverso.
-R	lista subdirectorios recursivamente, además del directorio actual.
-t	lista de acuerdo con la fecha de modificación de los archivos.

---

pwd [opción]
{Muestra el nombre del directorio de trabajo actual}

-P	muestra el directorio físico, sin enlaces simbólicos.

---

mkdir [opción]... directorio...
{Crea el/los directorio/s si no existe/n}

---

rmdir [opcion]... directorio...
{Borra el/los directorio/s si están vacíos}

-r	Borra recursivamente

---

touch [opción]... archivo...
{Se crea un archivo, si ya existían se ponen con la fecha y hora actual}

---

rm [opción]... archivo o directorio...
{Borra archivos y directorios con contenido}

-r	Borra recursivamente.

---

cat [opcion]... [archivo]...
{Concatena archivos y los imprime en pantalla}

-E	Imprime $ al final de cada línea.
-b	Enumera cada línea no vacía.
-n	Enumera cada línea.
-r	Copia recursivamente
-s	Suprime las líneas repetidas vacías.

---

cp [opción]... archivo o directorio... directorio
{Copia el primer archivo o directorio en el segundo directorio}

-s	crea un enlace en vez de copiar el archivo o directorio
-r	copia recursivamente

---

mv [opción]... archivo o directorio... directorio
{Mueve archivos}

bc -l
{Usa la librería matemática. Se usa con una pipe line delante.}

---

file [] archivo...
{Muestra los tipos de archivos}

-b	no muestra el nombre del archivo, sino que directamente dice el tipo.

---

head [opción]... [archivo]...
{Muestra la parte inicial de un archivo, por defecto 10 líneas}

--bytes=[-]K	imprime las primeras K bytes. (Con "-", imprime todo menos las ultimas k bytes).
--lines=[-]K	imprime las primeras K líneas. (Con "-", imprime todo menos las ultimas k líneas).

---

tail [opción]... [archivo]...]
{Muestra la parte final de un archivo, por defecto 10 líneas}

--bytes=K	imprime las últimas K bytes.
--lines=K	imprime las últimas K líneas.

---

sort [opción]... [archivo]...
{Ordena, según el criterio elegido, el contenido de los archivos}

Metacaracteres de archivo

?	Representa cualquier carácter simple en la posición en la que se indique.
*****	Representa cualquier secuencia de cero o más caracteres.
[ ]	Designan un carácter o rango de caracteres en cuyo caso, mostramos el primer y último carácter del rango separados por un guión "-".
{ }	Sustituyen conjuntos de palabras separadas por comas que tienen partes comunes.
~	Se usa para abreviar el camino absoluto (path) del directorio HOME.

---

chmod [opcion]... modo... archivo...
chmod [opcion]... modo octal... archivo...
{Cambia los permisos de acceso de los archivos- Si se usa el "+", otorga acceso. Si se usa el "-" quita acceso}

u	Propietario
g	Grupo
o	Resto de usuarios
a	Todos los grupos de usuarios
r	Lectura
w	Escritura
x	Ejecución

---

wc [opción]... [archivo]...
{Imprime el número de líneas, de palabras y de bytes de cada archivo junto a su nombre}

-c	Imprime el número de bytes
-m	Imprime el número de caracteres
-l	Imprime el número de líneas
-w	Imprime el número de palabras
-L	Imprime la longitud de la línea más larga

---

echo
{imprime una línea de texto}

---

date [opcion]... [+%formato]
{Imprime la fecha y la hora}

%a	imprime las siglas del día de la semana.
%A	imprime el día de la semana completo.
%b	imprime las siglas del mes.
%B	imprime el mes completo.
%c	cambia el orden de la salida, orden común español.
%C	imprime los dos primero dígitos del año
%d	imprime el día del mes
%D	imprime la fecha (lo mismo que %m/%d/%y)
%F	imprime la fecha completa (lo mismo que %Y-%d-%d)
%H	imprime la hora (00...23)
%I	imprime la hora (01...12)
%j	imprime el día del año
%m	imprime el numero del mes (01...12)
%M	imprime el mínuto (00...59)
%n	imprime una nueva línea
%r	imprime la hora (ej: 11:11:04 PM)
%R	imprime la hora y minutos (lo mismo que %H:%M)
%S	imprime los segundos (00...60)
%t	imprime una tabulación
%T	imprime la hora, minutos y segundos (lo mismo que %H:%M:%S)
%u	imprime el día de la semana de forma numérica
%V	imprime la semana del año
%W	imprime la semana del año (00...53)
%y	imprime los últimos dos dígitos del año
%Y	imprime el año

Metacaracteres de redirección

< nombre	Redirecciona la entrada de una orden para que la obtenga del archivo nombre .
> nombre	Redirige la salida de una orden para que la escriba en el archivo nombre . Si ya existe, lo sobreescribe.
&> nombre	La salida estándar se combina con la salida de error y se escriben en el archivo nombre .
>> nombre	Funciona igual que “>” pero añadiendo la salida al final de archivo, sin sobreescribir.
2> nombre	Redirige la salida de error a un archivo.
|	Crea un cauce entre dos órdenes. La salida de una de ellas se utiliza como entrada de la otra.
|&	Crea un cauce entre dos órdenes utilizando las salidas estándar y error de una como entrada de la otra.

Metacaracteres sintácticos

;	Separador entre órdenes que se ejecutan secuencialmente.
( )	Se usan para aislar órdenes separadas por “;” o por “|”. Las órdenes dentro de los paréntesis son tratadas como una única orden.
&&	Separador entre órdenes, en la que la orden que sigue al metacarácter “&&” se ejecuta sólo si la orden precedente ha tenido éxito (no ha habido errores).
| |	Separador entre órdenes, en la que la orden que sigue al metacarácter “||” se ejecuta sólo si la orden precedente falla.

---

env / printenv [opcion]...
{Permite visualizar las variables de entorno (o variables globales) comunes a todos los shells. Para estas variables se usan letras mayúsculas}
set
{Permite visualizar las variables locales, que solo son visibles en el shell donde se definen}

Asignacion de variables
Para asignar un valor a una variable se pone el nombre de la variable, un signo igual y el valor que deseamos asignar, que puede ser una constante u otra variable. A cada lado del signo igual no debe haber ningún espacio en blanco. Si delante o detrás del igual dejamos un espacio en blanco obtendremos un error, porque lo tomará como si fuera una orden y sus argumentos, y no como una variable. Además, el nombre de una variable puede contener dígitos pero no puede empezar por un dígito

---

Tipos de variables
- Cadenas: su valor es una secuencia de caracteres.
- Números: se podrán usar en operaciones aritméticas.
- Constantes: su valor no puede ser alterado.
- Vectores o arrays: conjunto de elementos a los cuales se puede acceder mediante un índice. Normalmente el índice es un número entero y el primer elemento es el 0.

Variables vectores
Para crear variables de tipo vector utilizamos la misma forma de definición, pero los elementos del vector se ponen entre paréntesis y separados por espacios. Para acceder a uno de sus elementos se pone ${variable[numero_de_elemento]}

$?
{Esta variable contiene el código de retorno de la última orden ejecutada, bien sea una instrucción o un guión.}

---

unset [variable ...]
{Borra las variables y sus atributos}

---

declare [-iarx] [-p] [variable[=valor] …]
{Crea variables con ciertos atributos}

-i	indica que la variable es numérica
-p	permite visualizar los atributos de la variable
-a	indica que es una matriz (vector o lista)
-r	indica que es de solo lectura
-x	indica que es exportable

---

export [-fn] [variable[=valor] …]
export -p
{Exporta las variables locales para que se puedan utilizar fuera del shell actual. Si se proporciona un valor, este se le asigna antes de exportarla}

-f	se refiere a funciones de shell
-n	borra la propiedad de exportación para la/s variable/s
-p	muestra una lista de todas las variables y funciones exportadas

---

printf “formato” [argumento]...
{Imprime los argumentos según el formato. Si pones un número entre el % y la letra de formato, se deja una separación antes del ultimo caracter de tantos caracteres como indique el número. Si el número es negativo, se deja la separación a partir del primer carácter con el siguiente arguento}

Formato:

\”	comillas dobles
\'	comilla simple
\\	barra invertida
\b	espacio atrás
\n	nueva línea
\t	tabulador
\0n	n=numero octal que representa un carácter ASCII de 8 bits
%d	un número con signo
%f	un número en coma flotante (decimal) sin notación exponencial
%q	entrecomilla una cadena
%s	Muestra una cadena sin entrecomillar
%x	Muestra un número en hexadecimal
%o	Muestra un número en octal

---

alias [-p] [nombre[='valor'] …]
{Define o muestra aliases. Dentro de un alias y entre comillas podemos poner varias órdenes separadas por “;” de tal forma que se ejecutarán cada una de ellas secuencialmente}

-p	Muestra todos los aliases definidos en un formato reusable

---

find [-H] [-L] [-P] [directorios...] [expresión]
{El comando find explora una rama de directorios buscando archivos que cumplan determinados criterios. Por defecto visualiza todos los archivos y directorios del directorio local y subdirectorios (incluso ocultos)}

-H	Solo sigue enlaces simbólicos cuando se están procesando los argumentos de la línea de comandos.
-L	Sigue los enlaces simbólicos.
-P	Nunca sigue los enlaces simbólicos.

Expresiones:

-name <u>nombre</u>	Busca por el nombre de archivo. Se pone -name seguido del nombre deseado
-atime <u>n</u>	Busca por el último acceso. Se pone -atime seguido de un número n, sin signo si quieres un numero de días exacto, con un + si quieres un número de días mayor al número y con un - si quieres un número de días menor al número.
-type d	Busca directorios.
-type f	Busca archivos regulares.
-size <u>n</u>	Busca archivos por tamaño en bloques. Si se pone sin signo, es el número exacto. Además se pueden incluir las letras c para buscar por bytes, k para buscar por kilobytes, M para buscar por Megabytes y G para buscar por Gygabytes.
-exec	Permite añadir una orden que se aplicará a los archivos localizados. La orden se situará a continuación de la opción y debe terminarse con un espacio, un carácter \ y a continuación un ;. Se utiliza {} para representar el nombre de archivos localizados.
-ok	Es similar a -exec, con la excepción de que solicita confirmación en cada archivo localizado antes de ejecutar la orden.

---

expr expresión
{Imprime el valor de la expresión, debe haber un espacio entre los argumentos y el operador}

Comillas

Dobles	Acotación débil. Protege cadenas desactivando el significado de los caracteres especiales que haya entre ellas, salvo los caracteres !, $, \ y `.
Simples	Acotación fuerte. Protege cadenas desactivando el significado de los caracteres especiales que haya entre ellas, salvo el caracter !.
Invertidas	Ejecuta las ordenes que se encuentren encerradas entre ellas e incluye, en el mismo sitio, el resultado que estas ordenes den. Se pueden sustituir por $(orden argumentos).

---

grep [opciones] patrón [archivo ...]
{Permite buscar cadenas en archivos utilizando patrones para especificar dicha cadena. Lee de una lista de archivos especificados como argumentos y escribe aquellas líneas que contengan la cadena.}

Opciones:

-x	Localiza líneas que coincidan totalmente, desde el principio hasta el final de línea, con el patrón especificado.
-v	Selecciona todas las líneas que no contengan el patrón especificado.
-c	Produce solamente un recuento de las líneas coincidentes.
-i	Ignora las distinciones entre mayúsculas y minúsculas.
-n	Añade el número de línea en el archivo fuente a la salida de las coincidencias.
-l	Selecciona sólo los nombres de aquellos archivos que coincidan con el patrón de búsqueda.
-e	Especial para el uso de múltiples patrones e incluso si el patrón comienza por el carácter (-).
-E	Toma la expresión como una expresión regular extendida. (egrep)
-F	Toma la expresión como una cadena literal escapando los símbolos. (fgrep)
-r	Lee todos los subdirectorios recursivamente. (rgrep)

Caracteres especiales:

[aeiou]	una vocal minúscula.
[A-Z0-9]	una letra mayúscula o una cifra.
[^0-9]	cualquier carácter que no sea una cifra.
*	indica que el elemento que le precede debe estar 0 o más veces.
.	concuerda con un carácter.
$	si aparece al final de la expresión significa final de línea.
^	si aparece al principio de la expresión significa principio de línea
\	elimina el significado especial al carácter que le sigue.

---

egrep patrón [archivo …]
{Es lo mismo que grep -E}

?	indica que el elemento que le precede debe estar cero o una vez.
+	indica que el elemento que le precede debe estar una o más veces.
{n}	indica que el elemento que le precede debe estar exactamente n veces.
{n,m}	indica que el elemento que le precede debe estar entre n y m veces.
(expr1 | expr2)	indica que puede aparecer expr1 o expr2

---

fgrep patrón [archivo ...]
{Es lo mismo que grep -F}

---

rgrep patrón [archivo ...]
{Es lo mismo que grep -r}

Variables

$0	Nombre del guión o script que se ha llamado. Sólo se emplea dentro del guión.
$1...$9...${n}	Son los distintos argumentos que se pueden facilitar al llamar a un guión. Los nueve primeros se referencian con $1, $2,..., $9, y a partir de ahí es necesario encerrar el número entre llaves, es decir, ${n}, para n>9.
$*	Contiene el nombre del guión y todos los argumentos que se le han dado. Cuando va entre comillas dobles es equivalente a “$1 $2 $3 ... $n”.
$@	Contiene el nombre del guión y todos los argumentos que se le han dado. Cuando va entre comillas dobles es equivalente a “$1” “$2” ... “${n}”.
$#	Contiene el número de argumentos que se han pasado al llamar al guión.
${arg:-val}	Si el argumento tiene valor y es no nulo, continua con su valor, en caso contrario se le asigna el valor indicado por val.
${arg:?val}	Si el argumento tiene valor y es no nulo, sustituye a su valor; en caso contrario, imprime el valor de val y sale del guión. Si val es omitida, imprime un mensaje indicando que el argumento es nulo o no está asignado.
$USER	Es el nombre de tu usuario
$IFS	“Internal Field Separator”. Determina como Bash reconoce los campos o límites cuando interpreta cadenas de caracteres. Por defecto es el espacio en blanco, tabulador o nueva linea, pero esto se puede cambiar.

Expresiones con variables

{El shell bash ofrece dos posibles sintaxis para manejar expresiones aritméticas haciendo uso de lo que se denomina expansión aritmética, o sustitución aritmética, que evalúa una expresión aritmética y sustituye el resultado de la expresión en el lugar donde se utiliza.}

$((...))
$[...]

Hay que tener en cuenta que las variables que se usen en una expresión aritmética no necesitan ir precedidas del símbolo $ para ser sustituidas por su valor, aunque si lo llevan no será causa de error, y que cualquier expresión aritmética puede contener otras expresiones aritméticas ya que se pueden anidar.

Operadores aritméticos

+ -	Suma y resta, o más unario y menos unario.
* / %	Multiplicación, división (truncando decimales), y resto de la división.
**	Potencia.
++	Incremento en una unidad. Puede ir como prefijo o como sufijo de una variable: si se usa como prefijo (++variable), primero se incrementa la variable y luego se hace lo que se desee con ella; si se utiliza como sufijo (variable++), primero se hace lo que se desee con la variable y después se incrementa su valor.
–	Decremento en una unidad. Actúa de forma análoga al caso anterior, pudiendo usarse como prefijo o como sufijo de una variable (--variable o variable--).
(...)	Agrupación para evaluar conjuntamente; permite indicar el orden en el que se evaluarán las subexpresiones o partes de una expresión.
,	Separador entre expresiones con evaluación secuencial.
=	x=expresión, asigna a x el resultado de evaluar la expresión (no puede haber huecos en blanco a los lados del símbolo “=”).
+= -=	x+=y equivale a x=x+y; x-=y equivale a x=x-y.
*= /=	x*=y equivale a x=x*y; x/=y equivale a x=x/y.
%=	x%=y equivale a x=x%y.

Operadores relacionales

A = B	A == B	A -eq B	A es igual a B
A != B	A -ne B		A es distinta de B
A < B	A -lt B		A es menor que B
A > B	A -gt B		A es mayor que B
A < =B	A -leB		A es menor o igual que B
A >= B	A -ge B		A es mayor o igual que B
! A			A es falsa
A && B	A -a B		A es verdadera y B es verdadera
A || B	A -o B		A es verdadera o B es verdadera

azul → Para números enteros
rojo → Para cadenas de caracteres

---

test expresión
[expresión **]**
{Esta orden evalúa una expresión condicional y da como salida el estado 0, en caso de que expresión se hayaevaluado como verdadera (true), o el estado 1, si la evaluación ha resultado falsa (false) o se le dio algúnargumento no válido.}

Expresiones:

-a archivo	archivo existe (solo para archivos, no directorios).
-b archivo	archivo existe y es un dispositivo de bloques.
-c archivo	archivo existe y es un dispositivo de caracteres.
-d archivo	archivo existe y es un directorio.
-f archivo	archivo existe y es un archivo plano o regular.
-G archivo	archivo existe y es propiedad del mismo grupo del usuario.
-h archivo	archivo existe y es un enlace simbólico.
-O archivo	archivo existe y es propiedad del usuario.
-s archivo	archivo existe y es no vacío.
-r archivo	archivo existe y el usuario tiene permiso de lectura sobre él.
-w archivo	archivo existe y el usuario tiene permiso de escritura sobre él.
-x archivo	archivo existe y el usuario tiene permiso de ejecución sobre él, o es un directorio y el usuario tiene permiso de búsqueda en él.
archivo1 -nt archivo2	archivo1 es más reciente que archivo 2, según la fecha de modificación, o si archivo1 existe y archivo2 no.
archivo1 -ot archivo2	archivo1 es más antiguo que archivo 2, según la fecha de modificación, o si archivo1 existe y archivo2 no.
archivo1 -ef archivo2	archivo1 es un enlace duro al archivo2, es decir, si ambos se refieren a los mismos número de dispositivo e inode.

Expresiones regulares

\	La barra de escape; si se quiere hacer referencia a este mismo carácter, debe ir precedido por él mismo y ambos entre comillas simples.
.	Cualquier carácter en la posición en la que se encuentre el punto cuando se usa en un patrón con otras cosas; si se usa solo, representa a cualquier cadena. Si se quiere buscar un punto como parte de un patrón, debe utilizarse \. entre comillas simples o dobles.
()	Un grupo; los caracteres que se pongan entre los paréntesis serán considerados conjuntamente como si fuesen un único carácter. (Hay que usar \)
?	Que el carácter o grupo al que sigue puede aparecer una vez o no aparecer ninguna vez. (Hay que usar \)
*	Que el carácter o grupo al que sigue puede no aparecer o aparecer varias veces seguidas. (No hay que usar \)
+	Que el carácter o grupo previo debe aparecer una o más veces seguidas.
{n}	Que el carácter o grupo previo debe aparecer exactamente n veces. (Hay que usar \)
{n, }	Que el carácter o grupo previo debe aparecer n veces o más seguidas. (Hay que usar \)
{n,m} pero	Que el carácter o grupo previo debe aparecer de n a m veces seguidas; al menos n veces, no más de m veces. (Hay que usar \)
[]	Una lista de caracteres que se tratan uno a uno como caracteres simples; si el primer carácter de la lista es "^", entonces representa a cualquier carácter que no esté en esa lista.
-	Un rango de caracteres cuando el guion no es el primero o el último en una lista; si el guion aparece el primero o el último de la lista, entonces se trata como él mismo, no como rango; en los rangos de caracteres, el orden es el alfabético, pero intercalando minúsculas y mayúsculas – es decir: aAbB...-; en los rangos de dígitos el orden es 012... También es posible describir rangos parciales omitiendo el inicio o el final del rango (por ejemplo [m-] representa el rango que va desde la "m" hasta la "z").
^	Indica el inicio de una línea; como se ha dicho anteriormente, cuando se usa al comienzo de una lista entre corchetes, representa a los caracteres que no están en esa lista. Situando a continuación de ^ un carácter, filtrará todas aquellas líneas que comiencen por ese carácter.
$	Indica el final de una línea. Situando un carácter antes del $, filtrará todas aquellas líneas que terminen por ese carácter.
\b	El final de una palabra. (Debe utilizarse entre comillas simples o dobles)
\B	Que no está al final de una palabra. (Debe utilizarse entre comillas simples o dobles)
\<	El comienzo de una palabra. (Debe utilizarse entre comillas simples o dobles)
\>	El final de una palabra. (Debe utilizarse entre comillas simples o dobles)
|	El operador OR para unir dos expresiones regulares, de forma que la expresión regular resultante representa a cualquier cadena que coincida con al menos una de las dos subexpresiones. (La expresión global debe ir entre comillas simples o dobles; además, cuando se usa con grep , esta orden debe iracompañada de la opción -E)

---

who
{Muestra quien está logueado en este momento.}

---

whoami
{Muestra el nombre del usuario. Es equivalente a "echo $USER".}

---

id opción... [usuario]
{Muestra la información de "usuario" y grupo para el usuario especificado, o (cuando se omite "usuario") para el usuario actual.}

-u	Número de usuario.
-g	Número del ID de grupo.
-n	Imprime el nombre en vez del número. Se ha de combinar con los anteriores.

---

cut opción... [archivo]...
{Imprime la parte de cada fila hasta el delimitador, entre dos delimitadores o caracteres concretos.}

--delimiter=*simbolo*	Establece un delimitador.
--fields=*num.columna*	Establece el número de columna a imprimir.
-s	Imprime solo las filas con el delimitador
--characters=*num*	Establece el número del carácter a imprimir de cada fila.

---

sed [opción]... [archivo]

-d	Borra las líneas con el patrón que le indiques.
'/./!d'	Borra las líneas vacías.

---

stat [opción]... archivo...
{Muestra el estado de un archivo.}

--format=*formato*	Añade formatos de salida

Formatos:

%x	Momento del último acceso.
%y	Momento de la última modificación.
%z	Momento del último cambio.
%U	Usuario del propietario del archivo.

Groups [username]...
{Imprime el grupo del cual es miembro cada username. Si no se especifica username, es para el proceso actual.}

---

for nombre [in lista]
do
    declaraciones que pueden usar $nombre
done
{Ejecuta una lista de declaraciones un número fijo de veces.}

EJEMPLO TIPO C++:
for i in `seq 1 1 5`
do
    declaraciones que pueden usar $nombre
done

IFS = `echo -e "\n\b"`
{Poner delante del for si utilizas listas para arreglar el problema de espacios.}

---

while
{Ejecuta una lista de declaraciones repetidamente mientras cierta condición se cumple.}

---

until
{Ejecuta una lista de declaraciones repetidamente hasta que se cumpla cierta condición.}

case expresión **in**
    patron1**)**
        declaraciones**;;**
    patron2**)**
        declaraciones**;;**
    …
        …
esac
{Ejecuta una de varias listas de declaraciones dependiendo del valor de una variable.}

---

read [-ers] [-a array] [-d delim] [-i text] [-n nchars] [-p prompt] [-t timeout] [-u fd] [name …]
{Lee desde el teclado}

-e
-r
-s
-a array
-d delimitador
-i text
-n nchars
-p prompt
-t timeout
-u fd

---

seq [valor inicial] [incremento] valor final
{Imprime los números desde el valor inicial hasta el valor final con el incremento asignado. Si no se asigna valor inicial o incremento, se toman por defecto ambos como 1}.