

Comandos Bash

COMANDOS BASICOS

mkdir <nuevo folder> / **mkdir** <folder1/folder2/.../nuevo folder> crear folder en la ruta actual / crear folder en la ruta especificada

cd <folder1/folder2/.../folder destino> cambiar de ubicación

cd .. subir un nivel referente a la ubicación actual

cd ~ regresar a la ruta "home"

cp <archivo> <archivo nuevo> copiar un archivo a archivo nuevo dentro de una ruta especificada, o no (en caso se quiera hacer en la misma ruta)

mv <archivo > <archivo nuevo> mover un archivo a archivo nuevo dentro de una ruta especificada, o no (en caso se quiera hacer en la misma ruta). Es similar al cp pero cuando se hace mv el archivo original no se mantiene

rm <ruta/archivo> elimina un folder especificado en una ruta o un archivo

history lista los comandos utilizados durante la sesión

cat <archivo> muestra el contenido del archivo

head -n <archivo> muestra las primeras n filas de un archivo especifico

tail -n <archivo> muestra las ultimas n filas de un archivo especifico. Se pueden usar ambos comandos (head/tail) para combinar su efecto. Por ejemplo, head -n1 <archivo | tail -n2 devuelve las ultimas n2 filas de las primeras n1 filas de un archivo

ls <ruta> devuelve el listado de elementos de la ruta especificada. Si no se especifica, devuelve el listado de la ruta actual

man <comando> muestra el manual de uso del comando especificado

grep <valor buscado> <archivo> selecciona las lineas de un archivo que tengan el valor buscado. Se pueden emplear las siguientes opciones: -v: invierte la búsqueda mostrando las lineas que no son iguales al valor. -c: muestra la cantidad de lineas que cumplen con la búsqueda -n: muestra los números de línea que cumplen con la búsqueda -h: no imprime los nombres de los archivos cuando se buscan múltiples archivos -l: imprime los nombres de los archivos que contienen la búsqueda -i: para tratar las mayúsculas y minúsculas como iguales (Algebra=algebra)

<línea de comando> > <archivo> el comando ">" guarda el resultado de la ejecución de un comando a un archivo

Comodines: * , ? * *: reemplaza todos los caracteres. Si coloca antes de una cadena reemplaza todos los caracteres desde el inicio has

COMANDOS SHELL

echo <\$variable/"cadena"> imprime en pantalla el valor de la variable o cadena que se coloca

#<cadena> se utiliza # para colocar un comentario dentro de una shell. Al momento de la ejecución lo que este a la derecha de este caracter se ignorará

ejemplos de bucle (for, while)

```
for anio in 2018 2017 2016;
```

```
do echo $anio;
```

```
done
```

```
for ((x=2;x<=12;x+=3))
```

```
do echo $x
```

```
done
```

```
x=1
```

```
while [ $x -le 12 ];
```

```
do
```

```
echo $x
```

```
((x+=3))
```

```
done
```

#usando wildcards:

```
for archivo in $files_ventas;
```

```
do echo $archivo;
```

```
done
```

bloques if:

```
if [<validacion1>] then
```

```
<bloque if 1>
```

```
elif (( validacion2 )) then #se pueden usar [] o (( ))
```

```
<bloque if 2>
```

```
else
```

```
<bloque else>
```

```
fi
```

```
fi
```

división de decimales: `echo $(echo "15/3600" | bc -l)` para realizar operaciones decimales se debe usar el comando "bc". Antes de ello, se debe instalar la librería necesaria con la siguiente línea: `sudo apt-get install bc`

\$1 \$2 .. \$n para capturar variables externas en nuestra shell, se usan "\$n" donde n es un numero entero desde 1 sucesivamente hasta n valores necesarios para nuestra lógica.



By **edgarchavc**

Published 5th May, 2020.

Last updated 5th May, 2020.

Page 1 of 2.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>