Sesión de terminal

Índice

Cuando comienza una sesión Linux inicia un entorno de usuario que es único para cada una de las cuentas de usuario. Varios atributos son establecidos y permanecen activos durante la sesión hasta que son cambiados. Algunas características que configuran el entorno de usuario son:

- **Directorio personal**: Identifica una localización en el disco duro del ordenador donde grabar y proteger los archivos del usuario. El directorio inicial de los usuarios suele incluirse dentro del directorio /home. Por ejemplo, si el nombre del usuario es Juan, su directorio personal será /home/Juan.
- **Una configuración de intérprete de comandos:** Hay varios intérpretes de comandos disponibles para usar con Linux, y cada uno tienes unas características. El intérprete de comandos bash (Bourne Again Shell) es el más usuado en Linux. Dentro del directorio personal hay varios ficheros de configuración que definen propiedades para la sesión de intérprete de comandos. Estos archivos deberán identificar las rutas, contienen variables de entorno y alias.
- Los archivos de configuración normalmente empiezan con un punto, y por lo tanto no aparecen por defecto cuando se lista el contenido del directorio. El comando ls -a permite ver dichos archivos.

Cómo utilizar el intérprete de comandos

Cuando se escribe un comando en el intérprete también se pueden incluir otros caracteres que cambien o añadan algo al funcionamiento del comando. Además del propio comando, éstos son algunos de los elementos que se pueden teclear en una línea de intérprete de comandos:

- **Opciones:** La mayoría de los comandos tienen una o más opciones que se pueden añadir para modificar su comportamiento. Generalmente las opciones consisten en una sola letra, precedida de un guión. Normalmente también se pueden combinar varias opciones tras un único guión. Por ejemplo, el comando ls -la enumera los contenidos del directorio actual. La -l pide una lista extensa de información, y la -a pide que también se enumeren todos los ficheros que empiecen por un punto. Cuando una sola opción se compone de una palabra o abreviatura, se le puede anteponer un doble guión (--). Por ejemplo, para utilizar la opción de ayuda de muchos comandos, escriba –help detrás del comando.
- **Argumentos:** Muchos comandos también aceptan argumentos después de introducir cualquier opción. Un argumento es un fragmento adicional de información, como un nombre de archivo, que el comando puede utilizar. Por ejemplo, cat /etc/passwd imprime los contenidos del fichero /etc/passwd, en este caso /etc/passwd es el argumento.
- **Variables de entorno:** El propio intérprete de comandos almacena información que puede ser de utilidad para el usuario de la sesión en lo que se denomina variables de entorno. Algunos ejemplos de variables de entorno son $SHELL (que identifica el intérprete de comandos), $SP1 (que define el símbolo de intérprete de comandos), etc.
- **Metacaracteres:** Son caracteres que tienen un significado especial para el intérprete de conandos. Los metacaracteres se pueden utilizar para dirigir la salida de un comando hacia un fichero (>), comunicar la salida a otro comando (|), o ejecutar un comando en segundo plano (&).

Cómo localizar comandos

Si se sabe dónde está localizado un comando en el sistema de archivos de Linux, una forma de ejecutarlo es escribir la ruta completa de ese comando. Por ejemplo, para ejecutar el comando date, ubicado en el directorio bin:

```
$ /bin/date
```

Como esto puede resultar pesado si el comando reside en un directorio de nombre largo, la mejor forma es tener almacenados comandos en directorios conocidos y después añadir esos directorios a la variable de entorno PATH del intérprete de comandos. El path (ruta) consiste en una lista de directorios que se comprueban secuencialmente para los comandos que se introducen. Para visualizar el contenido de PATH telclea:

```
$ echo $PATH
```

y mostrará algo parecido a esto: /bin:/usr/bin:/usr/local/bin:/usr/bin/X11:/home/usuario, que no es ni más ni menos que un listado de directorios separados por dos puntos.

El orden en la ruta de los directorios es importante. Los directorios se revisan de izquierda a derecha. Así, en este ejemplo, si hubiera un comando llamado EJECUTA en el directorio /bin, y otro comando llamado EJECUTA en el directorio /usr/bin, se ejecutará el que se encuentra en /bin.

Cómo conectar y expandir comandos

Una utilidad verdaderamente potente del intérprete de comandos es la capacidad de dirigir la entrada y la salida de conandos hacia y desde otros comandos y ficheros. Para permitir que los comandos se coordinen, el intérprete de comandos utiliza metacaracteres.

Comandos de redirección El metacaracter tubo (|) conecta la salida de un comando a la entrada de otro comando. Esto le permite tener un comando trabajando sobre unos datos y el siguiente comando ocupándose de los resultados. Ejemplo de una línea de comando que incluye tubierías:

$ cat /home/usuario/trabajos | sort | more

Este comando imprime los contenidos del fichero y conecta la salida con el comando sort. El comando sort toma los nombres de usuario con los que empieza cada línea del fichero, los ordena alfabéticamente y conceta la salida con el comando more. El comando more muestra la salida en una página cada vez.

Otro comando de redirección es el metacarácter > y >>

Comandos en segundo plano Algunos comandos pueden tardar en terminar y no se desea tener al intérprete de comandos esperando a que un comando termine. En estos casos, se pueden ejecutar comandos en segundo plano utilizando el signo &.

$ lpr documentomuygrande &

Expansión de comandos Lo explicaremos con un ejemplo:

$ vi $ (find /home -print | grep xyzzy)

en esta línea de comando se realiza, porque se ha indicado con $, primero la búsqueda de todos los ficheros cuyo nombre incluye xyzzy que se encuentran en el directorio /home. Después se pasan todos esos ficheros al comando vi que abrirá todos los archivos (uno cada vez) filtrados.

Con $ también pueden expandirse expresiones aritméticas, lo explicaremos de nuevo con un ejemplo:

$ echo “Yo tengo $[2005 – 1980] años”

mostrará en pantalla “Yo tengo 25 años”, y esto es así porque el intérprete de comandos realiza primero la expresión aritmética 2005-1980 y pasa esa información al comando echo.

Agrupamiento de órdenes Pueden utilizarse varias órdenes en una sola línea de las siguientes formas:

- Utilizando el punto y coma (;): ord1;ord2;ord3 provoca la ejecución sucesiva e independiente de todas las órdenes especificadas.
- Paréntesis: (ord1;ord2;ord3) tiene el mismo efecto que el caso anterior pero pueden considerarse como una única orden para ciertas actuaciones:

  (ord1;ord2;ord3) > archivo
- &&: ord1 && ord2: ejecuta la orden 2 sólo si la ejecución de la orden 1 no ha tenido éxito.
- ||: ord1 || ord2, ejecuta la orden 2 sólo si la ejecución de ord1 no ha tenido éxito.
- |: ord1 | ord2, ejecuta la orden 1 y encauza o envía su salida como entrada de ord2.

Comodines: El intérprete de comandos los utiliza para generar grupos o rangos de metacaracteres:

- ?: Sustituye a cualquier carácter, uno solo.
- *: Sustituye a cualquier carácter o grupo de caracteres.

- []: Sustituye cualquier valor incluido entre los corchetes. Admite rangos, ejemplo: [1-5], o inversiones en la selección cuando se precede del signo !, ejemplo: [!1-5]

Cómo utilizar las variables de entorno del intérprete de comandos

Todos los intérpretes de comandos activos almacenan la información que necesitan utilizando lo que se denominan variables de entorno. Una variable de entorno puede almacenar cosas como ubicaciones, ficheros de configuración, buzones de correo y rutas de directorios. También puede almacenar valores para los símbolos de intérpretes de conando, el tamaño del historial y el tipo de sistema operativo.

Para ver las variables de entorno que se encuentran asignadas actualmente en el intérprete de comandos se utiliza el comando declare.

Podemos referirnos al valor de cualquiera de las variables definidas poniendo delante del nombre de dicha variable el signo $, por ejemplo:

```
$ echo $PS1
```

nos mostrará en pantalla el valor de la variable PS1

Algunas variables de entorno del intérprete de comandos:

HOME: directorio de inicio. Es el directorio de trabajo actual. Es el valor que utiliza el comando cd para volver al directorio de trabajo desde cualquier otro directorio.

PATH: la lista de directorios separada por dos puntos que se utiliza para encontrar comandos ejecutables.

PS1: Establece el valor del símbolo de intérprete de comandos.

USER: nombre del usuario que ha abierto la sesión.

Estableciendo nuestras propias variables de entorno

Las variables de entorno pueden proporcionar una forma cómoda de guardar pequeñas cantidades de información que utilicemos a menudo.

Para establecer temporalmente una variable de entorno basta con teclear el nombre de la variable y asignarle un valor. Por ejemplo:

```
$ AB=/home/usuario/trabajo; export AB
```

en el ejemplo asignamos a la variable AB una ruta de directorio larga. Con el comando export propagamos el valor de la variable AB a cualquier otro intérprete de comandos que se pueda abrir.

Podremos utilizar la variable AB anteponiendo el signo $ al nombre: cd $AB será igual a cd / home/usuario/trabajo.

El problema de establecer variables de entorno de esta manera es que, tan pronto como salgamos del intérprete de comandos, se perderá su valor. Para ello habrá que modificar los ficheros de configuración de entorno que se estudian más adelante.

Una variable muy útil de actualizar es PATH, por ejemplo:

```
$ export PATH=$PATH:/home/usuario/scripts
```

se añade a las rutas de búsqueda de archivos ejecutables el directorio de usuario donde se guardan los scripts.

Cómo configurar el intérprete de comandos

Se puede ajustar el intérprete de comandos para que nos ayude a trabajar con mayor eficiencia. El símbolo de sistema puede suministrar información interesante cada vez que presionemos [Intro], se pueden establecer alias para salvar combinaciones de teclas y establecer permanentemente variables de entorno para adecuarse a nuestras necesidades.

Para que se guarden los valores y se obtengan cada vez que se inicia el intérprete de comandos, habrá que añadir esa información a los ficheros de configuración. Varios ficheros de configuración controlan el comportamiento del intérprete de comandos. Otros son específicos del usuario que crea el fichero de configuración. Mostraremos los ficheros que son de interés para todo usuario que utilice el intérprete de comandos bash en Linux:

/etc/profile Este fichero configura la información del entorno de usuario para cada usuario. Se ejecuta cuando se registra por primera vez y el intérprete de comandos se inicia. Este fichero proporciona valores por defecto para su ruta, su símbolo, para el tamaño máximo de los ficheros y los permisos que por defecto crea. También estabece las variables de entorno para cosas como la ubicación del buzón de correo y el tamaño del fichero de historial. Finalmente, etc/profile recopila los parámetros del intérprete de comandos de los archivos de configuración en el directorio /etc/profile.d

/etc/bashrc Este fichero se ejecuta para cada usuario que utiliza el intérprete de comandos bash. Se lee cada vez que se abre un intérprete de comandos bash. Establece el símbolo por defecto y puede añadir uno o más alias. Los valores de este fichero se pueden invalidar mediante la información contenida en el fichero HOME/.bashrc

HOME/.bash_profile Este fichero lo utiliza cada usuario para introducir información que es específica del uso propio del intérprete de comandos. Se ejecuta sólo una vez, cuando el usuario entre en el sistema. Establece por defecto variables de entorno y ejecuta el fichero . bashrc del usuario.

HOME/.bashrc Este fichero contiene la información sobre bash específica del interprete de comandos bash. Se lee cuando entra en el sistema y también cada vez que el usuario abre un nuevo intérprete de comandos bash. Es la mejor ubicación parfa añadir variables de entorno y alias para que el intérprete de comandos pueda cogerlos.

HOME/.bash_logout Este fichero se ejecuta cada vez que el usuario sale del último intérprete de comando bash.

Para cambiar los ficheros /etc/profile o /etc/bashrc hay que ser usuario root. Cualquier usuario puede cambiar la información contenido en los ficheros HOME/.bash_profile, HOME/.basrhc y HOME/.bash_logout

Cómo configurar el símbolo El símbolo de sistema (PROMPT) consiste en un conjunto de caracteres que aparecen cada vez que el intérprete de comandos está listo para aceptar un comando. Lo que contiene ese símbolo lo determina la variable de entorno PS1.

Algunos de los valores que puede tomar dicha variable son:

\$: muestra el símbolo de usuario estándar $.

\\: muestra una barra invertida.

\w: muestr la ruta completa del directorio de trabajo actual.

\u: muestra el nombre de usuario actual.

\t: muestra la hora actual en horas, minutos y segundos.

Por ejemplo:

```
$ export PS1="[\t\w]\$"
```

haría que el PROMPT tuviera el aspecto:

[20:26:30 /usr/bin] $

Para realizar un cambio permanente en el símbolo del sistema se debe añadir el valor de PS1 al fichero .bashrc. Probablemente ya hay un valor PS1 que se puede modificar.

Cómo añadir variables de entorno Se pueden añadir variables de entorno al fichero .bashrc para que el valor se mantenga entre sesiones.

Cómo añadir alias Establecer alias puede ahorrarnos más escritura que, incluso, establecer variables de entorno. Gracias a los alias podemos hacer que una cadena de caracteres ejecute toda una línea de comando. Podemos añadir y listar alias con el comando alias:

```
$ alias p='pwd ; ls – CF'
$ alias rm='rm -i'
```

En el primer ejemplo, se asigna la letra p para que ejecute el comando pwd, y depués para ejecutar ls -CF para mostrar el directorio actual de trabajo y listar sus contenidos en forma de columnas. El segundo ejecuta el comando rm con la opción -i cada vez que escriba rm (este alias a menudo se establece automáticamente para el usuario root, de forma que, en lugar de eliminar los ficheros, se le pregunte para cada eliminación individual. Esto evita que elimine todos los ficheros de un directorio al escribir por error algo como rm *)

Mientras nos encontramos en el intérprete de comandos, podemos comprobar qué alias están establecidos escribiendo el comando alias. Si se desea eliminar un alias utilizaremos el comando unalias.

Podemos hacer permanetes nuestros alias incluyéndolos en el fichero .bashrc.

Cómo trabajar con el sistema de archivos de Linux

El sistema de archivos de Linux es la estructura con la que se almacena toda la información en el ordenador. Los ficheros se organizan según una jerarquía de directorios. Cada directorio puede contener ficheros, así como otros directorios (La jerarquía de ficheros de Linux se tratará con mayor profundidad en temas posteriores, ésta es simplemente una primera pincelada que nos permita comprender los comandos ls, cp, mv, etc.).

Si se tuviera que representar mediante un dibujo los ficheros y directorios en Linux tendría el aspecto de un árbol boca abajo. En la parte superior está el directorio raíz, que se representa con una barra lateral (/). Debajo de él, encontramos un conjunto de directorios comunes en el sistema Linux, como por ejemplo: /bin, /dev, /home, /lib y /tmp, por nombrar algunos.

En el directorio /home se encuentran los directorios asignados a cada usuario.

Aunque son similares en muchos aspectos, el sistema de archivos de Linux presenta diferencias respecto a los sistemas de archivos utilizados en MS-DOS y Windows. Algunas de estas diferencias son:

- En el sistema de archivos de MS-DOS y Windows, las letras de unidad representan diferentes dispositivos de almacenaje (A:--> disquetera, C:-->disco duro). En Linux todos los dispositivos de almacenaje se encuentran dentro de la estructura de directorios. (Esto se verá con más detalle cuando tratemos los comandos mount y umount en temas posteriores).
- En Linux se utilizan barras laterales en lugar de barras invertidas para separar directorios. Así C:\home\usuario en un sistema Microsoft es /home/usuario en un sistema Linux.
- Los nombres de archivo tienen casi siempre en DOS un sufijo (.txt para texto, .doc para documentos word, etc) Si bien pueden utilizarse lo mismo en Linux, los sufijos de tres caracteres no tienen un significado en Linux, aunque pueden utilizarse para identificar un tipo de archivos.
- Cada archivo y directorio de un sistema Linux tiene permisos y propietarios asociados, cosa que se añadió con posterioridad a los sistemas Microsoft.

Para trabajar en el árbol de directorios del sistema Linux trabajamos en clase los siguientes comandos:

- cd: Cambia a otro directorio
- pwd: Muestra el nombre del directorio actual
- mkdir: crea un directorios
- rmdir: borra un directorio
- ls: lista los contenidos de un directorio. En este comando hemos conocido las opciones -l y -a.

Un directorio se puede referenciar de dos maneras posibles:

- absoluta: cuando le nombramos desde el directorio raíz, por ejemplo: /home/usuario/TRABAJO/COPIAS. Tiene la ventaja de ser siempre el mismo nombre nos encontremos en el directorio que nos encontremos.
- Relativa: cuando le nombramos utilizamos la referencia al directorio de trabajo actual, por ejemplo, si nos encontramos en el directorio /home/usuario/TRABAJO/INFORMES y queremos ir al COPIAS del párrafo anterior debemos trabajar con ../COPIAS.

En el sistema de archivos el . Indica el directorio actual, y los .. indican el directorio inmediatamente superior al directorio actual. Por ejemplo, si tecleamos el comando cd .. desde el directorio /home/usuario nos situaremos en el directorio /home.

Dentro de los directorios, y donde realmente se almacena la información, se encuentran los archivos. Como usuarios podemos crear, visualizar, copiar, mover, borrar y modificar los archivos, para ello hemos visto en

clase los siguientes comandos:

- cp <archivo1> <archivo2>: orden que realiza una copia del archivo1 en archivo2
- mv <archivo1> <archivo2>: orden que mueve el archivo1 a archivo2. En este caso el archivo1 desaparece
- rm <archivo>: elimina el archivo
- cat <archivo>: visualiza en pantalla el contenido del archivo
- sort <archivo>: ordena las líneas del archivo alfabéticamente
- wc <archivo>: contabiliza el número de líneas (opción -l), palabras (opción -w) y caracteres (opción -c) del archivo
- nl <archivo>: muestra el archivo con sus líneas numeradas
- grep “cadena” <archivo>: busca la “cadena” en el archivo y muestra en pantalla las líneas que contienen dicha cadena
- ln <archivo> <enlace>: Crea un enlace al archivo, lo que permite utilizar el nombre de enlace como si del mismo fichero se tratara.

Cómo crear archivos de texto

Trabajamos en clase tres posibilidades:

- Redireccionamiento, mediante el fichero **cat**, de la salida a pantalla.
- Mediante el comando **touch**.
- Mediante el editor de texto **vi**.

Con la primera opción comprobamos que se puede crear un fichero llamado trabajos.txt introduciendo el siguiente comando en la línea de órdenes del intérprete:

```
$ cat > trabajos.txt
```

A partir de aquí, todo lo que escribamos se mostrará en el terminal y quedará a su vez almacenado en el fichero trabajos.txt.

Para terminar el fichero basta con pulsar las teclas Control y C simultáneamente.

Si quisiéramos añadir nuevas líneas al fichero trabajos.txt utilizaríamos la opción de redireccionamiento >>.

Este método tiene el inconveniente de la imposibilidad de modificar el contenido de un texto ya escrito.

La segunda opción permite la creación de ficheros sin ningún contenido. Nos puede resultar útil, en nuestro caso, para la comprobación de distintas actividades planteadas.

La tercera opción es la más eficaz aunque la más compleja: el editor de texto vi. En clase trabajaremos, en este tema, exclusivamente el modo inserción y en el modo comando w(escribe el contenido del fichero), q (sale del editor vi), y la combinación de ambos wq (escribe y sale del editor). Para evitar problemas con los “buffers” que mantienen en memoria el editor, todos estos comandos los terminaremos con el signo !, quien permite la grabación y salida inmediata del editor.

Recordamos que para cambiar de modo edición a modo comando (y viceversa) se utiliza la tecla ESC, y que para trabajar con los comandos antes descritos hay que introducir previamente los dos puntos (:).

Cómo crear archivos ejecutables en el intérprete de comandos (scripts)

El intérprete de comandos puede programarse con su propio lenguaje de programación y, aunque este asunto lo trataremos con más detalle en un tema posterior, puede resultarnos útil la creación de pequeños archivos ejecutables que, a partir de ahora, denominaremos scripts.

En el punto anterior hemos visto la forma de crear un archivo texto. Esos archivos de texto pueden contener secuencias de comandos del intérprete.

Una vez creado el fichero bastará con ejecutar el comando:

```
$ chmod +x <nombre del fichero>
```

para que ese fichero pueda ejecutarse, e introduciendo <nombre del fichero> en la línea del intérprete de comandos realice toda la secuencia de instrucciones que hayamos indicado.

Otros comandos trabajados en clase

- who: Muestra en pantalla la relación de usuarios conectados actualmente a nuestro sistema
- tty: muestra el controlador de terminal asignado y su ruta de acceso. En los sistemas Linux es el fichero /dev/pts<número>, que no es ni màs ni menos que un fichero que puede ser manipulado con los derechos necesarios. En clase hemos realizado prácticas de redireccionamiento con el comando cat a dichos ficheros
- echo: muestra en pantalla el mensaje indicado a continuación o el valor de una variable de entorno si se precede a ésta del carácter $.
- clear: limpia el terminal de caracteres
- uname: muestra el nombre del sistema operativo
- exit: provoca la salida del terminal
- date: muestra la fecha y hora actuales
- cal: muestra un calendario en pantalla
- man: muestra información detallada sobre la orden especificada a continuación
- wc: contabiliza el número de líneas, palabras y/o caracteres de un archivo.

Actividades

1.- Muestra en la pantalla la fecha y hora actuales.

2.- Obtén el calendario de junio de 1995. Obtén el calendario de junio de 1995 con el lunes como primer día de la semana.

3.- Ejecuta las siguientes órdenes y observa el resultado:

```
echo date
echo 'date'
echo `date`
```

4.-Muestra en pantalla el calendario completo del año 2000. Vuelve a obtenerlo con la orden **cal 2000 | more**.

5.- Propón una expresión con comodines que incluya todos los nombres terminados en txt.

6.- Crea una expresión con comodines que incluya los nombres de fichero que contengan el carácter a o el carácter b.

7.- Construye una expresión con comodines que incluya los archivos equipoA, equipoB, ... hasta equipoM.

8.- Elabora una expresión con comodines que incluya las palabras Cap12, Cap13, hasta Cap19.

9.- Propón una expresión con comodines que comprenda nombres que comiencen por un dígito seguido por tres caracteres cualesquiera, un signo + y cualquier terminación.

10.- Ejecuta las siguientes líneas de órdenes:

```
a) who ; tty            e) who tty
b) who && tty           f) who || tty
c) wha && tty           g) wha || tty
d) who && ttih) who || tti
```

11.- Crea la siguiente línea de órdenes y explica su efecto:

```
echo PATH $PATH /$PATH
```

12.- Genera con la orden echo el mensaje 2*4=8.

13.- Escribe textualmente la siguiente secuencia (sin incluir el prompt principal $ ni el secundario > que presenta automáticamente el shell):

```
$ echo “línea uno [Intro]
>línea dos [Intro]
>línea tres” [Intro]
```

14.- Repite lo hecho en la práctica 13 pero con comillas simples. ¿Qué ocurre?.

15.- Comprueba cuál es el directorio actual.

16.- Crea en tu directorio de trabajo los subdirectorios PRUEBA1 y PRUEBA2, y en este último el directorio PRUEBA2.1 Construye la orden en una sola línea. Comprueba el resultado.

17.- Convierte el directorio PRUEBA2.1 en el directorio actual. Compruébalo.

18.- Sitúate en el directorio raíz.

19.- Vuelve a tu directorio HOME y elimina el directorio PRUEBA2.1

20.- Desde el directorio HOME visualiza el contenido del directorio raíz.

21.- Modifica la orden de la actividad 20 para mostrar información detallada de cada archivo.

22.- Sitúate en el directorio HOME y comprueba si las siguientes órdenes son equivalentes:

```
cd /etc
cd ../../etc
cd
```

23.- Colócate en tu directorio HOME. Crea, a partir de él, la siguiente estructura de directorios:

```
TRABAJOS
        TEXTOS
        MODELOS
                IMPRESOS
```

INSTANCIAS
CIRCULARES
INFORMES

COPIAS
TEMPORAL

24.- Intenta crear en el directorio COPIAS dos directorios con el mismo nombre: INF.
25.- Trata ahora de crearlos en distintos directorios: COPIAS y TEMPORAL.
26.- Sitúate en el directorio INFORMES. ¿A qué directorio te llevarán las siguientes órdenes:

cd ..
cd ../../
cd ../IMPRESOS
cd ../../TEXTOS

(Se supone que tras cada orden regresamos al directorio INFORMES)
27.- Indica una orden con una ruta absoluta y otra con una ruta relativa para pasar del directorio CIRCULARES al directorio INSTANCIAS.
28.- Elimina la estructura de directorios de la actividad 23.
29.- Crea un directorio .INVISIBLE. Trata de descubrilo con la orden ls.
30.- Prueba a descubrir el directorio de la actividad 29 con la orden ls -a. ¿Podrías explicar por qué ocurre esto?
31.- Crea en el directorio PRUEBA1 un directorio datos1 de 4 líneas de texto con el nombre, apellidos, dirección, teléfono. Visualizaló en pantalla.
32.- Añade al fichero datos1 la fecha del día de creación.
33.- Haz una copia del archivo datos1 con el nombre datos2.
34.- Crea un enlace lógico del archivo datos1 con nombre datos3. Visualiza datos3.
35.- Añade una línea más al archivo datos1. Visualiza los archivos datos1, datos2 y datos3. ¿Qué ficheros coinciden en el contenido?
36.- Copia el archivo datos3 con el nombre datos4. ¿Qué relación habrá entre datos1 y datos4: copia o enlace lógico? ¿Podrías comprobarlo?¿Cómo?
37.- Mueve el archivo datos2 al directorio PRUEBA2.
38.- Mueve el archivo datos3 al directorio PRUEBA2 con el nombre datos5. ¿Qué relación habrá entre datos1 y datos5: copia o enlace lógico? ¿Podrías comproblarlo?¿Cómo?
39.- Cambia el nombre de datos4 por el de datos6.
40.- Añade todo el contenido de datos2 a datos1. Visualiza el nuevo contenido de datos1 y datos5.
41.- Añade el contenido (sin utilizar ficheros auxiliares) del fichero datos1 a datos1 ¿Puedes?. Añade el contenido de datos5 a datos1 ¿Puedes? Añade el contenido de datos6 a datos1.
42.- Contabiliza el número de líneas, palabras y caracteres del fichero datos1.
43.- Muestra en pantalla las líneas de datos1 ordenadas por orden alfabético.
44.- Crea el fichero datos1.num que contiene las líneas de datos1 numeradas.
45.- Ordena el fichero datos1 (realízalo en una sola línea y sin utilizar ficheros auxiliares).
46.- Muestra las líneas de datos1 que contenga tu nombre.
47.- Crea un archivo indice que contenga la lista de archivos (sin directorios) del directorio activo.
48.- Crea un archivo indice.ord que contenga la lista anterior numerada.
49.- Crea un subdirectorio denominado documentos. En este directorio un archivo agenda con nombre, dirección y teléfono de varias personas. Separa los campos con tabuladores. Visualiza la agenda ordenada por nombre.
50.- Visualiza la agenda ordenada por dirección.
51.- Visualiza la agenda ordenada por teléfono.
52.- Ordena la agenda anterior en orden inverso.
Nota: la opción +n de la orden sort permite realizar la ordenación a partir del campo n. La opción -r de la orden sort permite la ordenacdión inversa.