Preparación para el examen LPI 101

Tema 103.2

Procesando cadenas de texto usando filtros

Créditos y licencia de uso

Coordinación:
Manuel Guillán (xLekOx) lpi@xlekox.org
Oscar Casal (ocs) oscar@glug.es

Traducción:
Miguel Castiella (ruton) mcastiella@euskalnet.net
Carmen Eugenio (nemrac) meneiro@ono.com
Manuel Guillán (xLekOx) lpi@xlekox.org
Al H. Carril (Mandraker) spoofff@hotmail.com
Daniel Perelló (Baooroo) daniel_perello@hotmail.com

Maquetación:
Oscar Casal (ocs) oscar@glug.es
Rafael Díaz (fistipaldi) fistipaldi@teleline.es
Manuel Guillán (xLekOx) lpi@xlekox.org
Kiefer Von Jammo (Kiefer) khrooon@gmail.com

Distribuido por FreeUOC (www.freeuoc.org) bajo licencia: Attribution-NonCommercial-ShareAlike2.0 de commons creative

http://creativecommons.org/licenses/by-nc-sa/2.0/

ÍNDICE

Índice de contenido

Tema 103.2

Introducción

Este capítulo cubre la mayoría de las herramientas de procesado de textos disponibles en sistemas Linux. Éstas incluyen diversas utilidades de filtrado, que se usan para buscar y cambiar archivos, así como las herramientas de entrada y salida. Es necesario comprender el uso de estas herramientas, debido a que son especialmente útiles en las labores administrativas diarias. Por ejemplo, usando la herramienta sed para buscar y reemplazar texto en scripts, nos permite hacer cambios de un modo mucho más sencillo de lo que sería si tuviésemos que hacerlos de forma manual. Por otro lado, aparecerán seguro entre las preguntas del examen, así que asegúrate de entender bien las diferencias y usos de cada herramienta.

Los comandos que se verán en este tema son:

- cat
- cut
- expand
- fmt
- head
- join
- nl
- od
- paste
- pr
- sed
- sort
- split
- tac
- tail
- tr
- unexpand
- uniq
- wc

Así mismo se harán ejercicios sobre los mismos al final del tema, que serán muy parecidos a los realizados en los exámenes.

Este tema tiene un peso (importancia) de 6 de cara al examen final de la certificación LPI 101. El total de la suma de pesos de todos los temas es de 106.

GNU y comandos Linux

Procesa cadenas de texto usando filtros de procesamiento de textos. Envía ficheros de texto y produce cadenas a través de utilidades de filtro de textos para modificar la salida de una manera más útil. Incluye el uso de comandos estándar de Unix que se encuentran en el paquete de utilidades de texto (textutils) GNU como sed, sort, cut, expand, fmt, head, join, nl, od, paste, pr, split, tac, tail, tr, y wc.

Linux ofrece una variedad de herramientas para usar, procesar y filtrar texto. Estas herramientas permiten buscar datos y manipularlos dependiendo de cual se use. Las herramientas pueden usarse desde la línea de comandos o localizadas en scripts, las cuales se ejecutan para realizar las tareas necesarias.

Cuando trabaje con ficheros de texto preferirá usar un paginador como more o less. Estas utilidades son usadas para mostrar el texto página a página. La utilidad cat puede usarse también para mostrar el contenido de un fichero de texto. Estas utilidades no son filtros pero a menudo son muy útiles cuando se trabaja con ellos.

Listando el contenido de un fichero

Un comando muy usado para listar los contenidos de ficheros de configuración o ficheros de pruebas de tamaño pequeño es el comando cat, un ejemplo sencillo de su uso sería listar un fichero de prueba llamado nombres.txt con la orden:

```
$ cat nombres.txt
```

saldría por pantalla el contenido del fichero.

Ordenando lineas de un fichero

La utilidad sort ordena las lineas de un texto y las muestra en la salida estándar. Se usa para ordenar, mezclar, y comparar líneas de ficheros o de la entrada estándar. Esta utilidad se usa con la siguiente sintaxis:

```
sort -opción nombre_fichero
```

Tabla 2-1 muestra las opciones más usadas con la utilidad sort.

Opción	**Uso**
-b	Ignora blancos
-c	Revisa si el fichero está ordenado
-d	Considera solo los caracteres alfanuméricos y ordena por directorio teléfono
-f	Ignora si los caracteres están en minúsculas o mayúsculas
-m	Une los ficheros ya ordenados sin reordenarlos
-M	Compara ficheros ordenados
-n	Ordena numéricamente
-o FILE	Escribe en un fichero específico de salida en lugar de la salida estándar
-r	Invierte los resultados
--help	Muestra la ayuda y salida
--version	Muestra versión y salida

Sigamos con un ejemplo del comando sort

Creamos un fichero de texto llamado list con las siguientes líneas:

```
3
eggs
5
Celestial Seasons
bread
32
Whiskas
butter
41
milk
ice
Centrum
90
flour
sugar
```

Ejecutamos el comando

```
$ sort list
```

y observamos el resultado, por defecto, se ordena la lista por orden numérico seguido de orden alfabético, con mayúsculas antes que las minúsculas. En el siguiente ejemplo, la opción -f hace que sort ignore mayúsculas:

```
$ sort -f list
```

En este otro ejemplo, dos ficheros se fusionan y se ordenan. El primer archivo sería por ejemplo el archivo prueba1 con el siguiente contenido:

angie
birdgrlmon
kim
monica
denise
lisa
desteve
jcfraggle
nikks
judy
netchickie

y el otro, prueba2 con este otro contenido:

katrina
crystalmoon
lma
loudhouse
trinityz
jill
nikki

```
$ sort prueba1 prueba2
```

Si esos dos archivos estuvieran ordenados se podría usar el comando:

```
$ sort -m prueba1 prueba2,
```

su salida sería una combinación de los 2 archivos sin reordenación.

También es posible ordenar ficheros por campos. Los campos se pueden separar por espacios o tabuladores y son numerados empezando por cero. Cuando se ordenan campos, el símbolo + precede al numero de campo con cada fichero separado por espacios.

El fichero nameslist contiene la siguiente lista:

Scott Bessler
Jason Nash
Angie Nash
Derek Stutsman
Jeff Arellano
Paul Ward
Alex Blauvelt
Peter Anapol
David Goolsby
Michael Craig
Johannes Erdfelt
Thomas McCanta
Joakim Erdfelt
Pete Gizzi
Neil Schroeder

En el siguiente ejemplo, el fichero nameslist se ordena por el segundo campo (+1) y entonces por el primer campo (+0):

```
$ sort +1 +0 nameslist
```

El uso de estos campos permite mucha flexibilidad en ordenar listas en ficheros. Es importante recordar que la utilidad sort no cambia el fichero original. La salida se envía a la salida estándar donde se puede visualizar o redireccionar hacia otro comando o fichero.

Cortando texto

La utilidad cut se usa para escribir partes seleccionadas de un fichero en la salida estándar. La utilidad cut también se puede usar para seleccionar columnas o campos desde ficheros específicos. Es posible seleccionar un trozo de una línea específica, varios trozos, o un rango. La tabla 2-2 cubre las diversas opciones usadas con la utilidad cut.

Tabla 2-2 : Opciones usadas con cut

Opción	**Uso**
-b	Escribe en la salida solo el rango de bytes especificados
-c	Escribe en la salida solo los caracteres especificados
-f	Escribe en la salida solo los campos especificados, delimitados por tabuladores
-help	Muestra ayuda y luego sale
-version	Muestra información de la versión y luego sale

A continuación un ejemplo del uso de estos rangos para escribir en la salida solo los primeros diez caracteres de cada línea:

```
$ cut -c 1-10 nameslist
```

La utilidad cut puede especificar que se muestren bytes, caracteres o campos de ficheros.

Pegando texto

La utilidad paste permite juntar texto desde múltiples ficheros. Las líneas correspondientes del fichero específico se escriben en la salida estándar con cada línea separada por un carácter tabulador. Las opciones usadas con la utilidad paste se muestran en la tabla 2-3.

Tabla 2-3 : Opciones usadas con paste

Opción	**Uso**
-s	Pega líneas desde un fichero a la vez
-d delimit-list	Usa los caracteres especificados en delimit-list consecutivamente en vez del carácter tab cuando separa ficheros mezclados.

El contenido del fichero nicks es el siguiente:

Banmage
Rexmortis
Jackyl
Dragonstr
Alchemist
Just_joe
Johan
Sting
Dave
Thomas
Netchick
Netjunkie
Pri
Zaphod
Lordram

A continuación un ejemplo del uso de la utilidad paste,

```
$ paste nameslist nicks
```

Nos daría como resultado los nombres seguidos de los nicks, suponiendo que estuvieran ordenados los archivos del mismo modo y cada línea perteneciese a una única persona, este sería el resultado esperado teniendo en cuenta los ficheros fuente, sin embargo la utilidad paste une linea a linea cualquier fichero, esté o no ordenado y se haga corresponder o no.

Convirtiendo tabuladores en espacios

Cuando se crea un fichero en GNU/Linux, se usa la tecla tab para justificar el texto, pero según la configuración del sistema/editor/entorno, el tamaño de esta justificación puede variar.
Esto puede cambiar el aspecto de un fichero se vea en un SO o en otro (o en un entorno o en otro). Para hacer que el fichero se vea igual sin importar la configuración de esta justificación, se pueden cambiar las 'tabulaciones' por un numero especifico de espacios. Esto causará que el fichero se vea igual en todos los sistemas.

La utilidad expand se usa para expandir los tabuladores en espacios. Esta utilidad puede trabajar con texto tanto de un fichero como de una entrada estándar. Las opciones para la utilidad expand se muestran en la tabla 2-4.

Tabla 2-4 : Opciones utilizadas con expand

Opción	**Uso**
-I	Convierte sólo los tabuladores de comienzo de la línea
-t	Utiliza una número para especificar los espacios a convertir
--help	Muestra la ayuda
--version	Muestra la información de la versión

La acción por defecto de la utilidad expand es convertir todos los tabuladores de un fichero en ocho espacios.

El contenido del fichero ejemplo3 es el siguiente: (Entre el nombre y la F o la M hay un tabulador de separación, como se ve a simple vista no queda con el aspecto de una tabla corriente).

```
Scott Bessler	F
Jason Nash	M
Angie Nash	M
Derek Stutsmann		M
Jeff Arellano	S
Paul Ward	M
Alex Blauvelt	S
Peter Anapol	M
David Goolsby		S
```

Al ejecutar el comando

```
$ expand ejemplo3
```

Alineamos el contenido del archivo de modo que las M's y las F's queden a la misma altura, si la opción por defecto no es suficiente podemos usar la opción -t con el número de espacios que queremos substituir el tabulador, con el fin de poner todas las filas a la misma altura:

```
$ expand -t 15 ejemplo3
```

Convirtiendo espacios en tabuladores

La utilidad unexpand realiza la función inversa que expand, de modo que los espacios se convierten en tabuladores, las opciones más comunes se pueden consultar en su manual:

```
$ man unexpand.
```

Formateando párrafos.

La utilidad fmt formatea cada párrafo en un fichero y envía la salida al output estándar. Este comando es utilizado para especificar la anchura de las líneas y junta o separa líneas en un esfuerzo para que estas tengan la misma longitud. El comando fmt intenta separar líneas al final de cada sentencia. Cuando esto no es posible, intenta romper la línea después de la primera palabra o antes de la última palabra de la sentencia. Las sentencias finalizan con una marca de periodo, exclamación o cuestión (.!?) seguidas de dos espacios o un retorno de carro (line feed). Se lee todo el párrafo antes de que se introduzca cualquier salto de línea.

La anchura por defecto que utiliza fmt para una línea es de 75 caracteres. El ancho por defecto puede ser modificado usando la opción adecuada en el comando fmt. La tabla 2-5 recoge algunos de los parámetros del comando.

Tabla 2-5: Opciones utilizadas con fmt

Opción	**Uso**
-c	Mantiene igual los espacios de principio de párrafo y alinea el párrafo con margen izquierda en la segunda línea
-t	Trabaja como -c excepto que los espacios de comienzo de la segunda línea sean igual que la primera, considerando la segunda línea como un párrafo de una línea
-s	Especifica que las líneas van a ser divididas y no juntadas
-u	Especifica que espacio uniforme se va a utilizar; esto reduce la separación entre todas las palabras a un espacio y a dos espacios entre las sentencias
-NUMERO o -w NUMERO	Establece la largura de la línea al NUMERO indicado
-p PREFIX	Especifica que las líneas que empiecen por PREFIX serán modificadas

El comando fmt deja igual las lineas en blanco, los espacios de principio de párrafo y el spacing, así cualquier formateo especial dentro del documento no es modificado. Abajo mostramos un ejemplo del comando fmt. Como puedes ver en el ejemplo, todas las líneas entre párrafos son mantenidas, y el comando hace que todas las líneas sean de 40.

El fichero de ejemplo (ejemplo4) contiene una única línea con el siguiente texto:

Linux offers a variety of tools to use for pricessing and filtering text. These tools enable you to search for data...

Con el comando

```
$ fmt -40 ejemplo4
```

se formatea dicho texto.

Borrando o sustituyendo caracteres

Hay veces en las que se quiere buscar en un documento caracteres específicos y luego borrarlos o reemplazarlos por otros. Un ejemplo sería un documento que utiliza mayúsculas y minúsculas, pero se prefiere que todo el documento esté en minúsculas. El comando tr hace esto utilizando la siguiente síntesis:

```
tr option set1 set2
```

Las opciones que se utilizan con tr están en la tabla 2-6

Tabla 2-6 : Opciones utilizadas con tr

Opción	**Uso**
-d	Borra un carácter especificado.
-s	Remplaza una secuencia de caracteres por un carácter.
--help	Muestra la ayuda
--version	Muestra la versión

Esto es un ejemplo del comando tr. Primero, el contenido del fichero es mostrado utilizando el comando more. El documento tiene una mezcla de mayúsculas y minúsculas. Todos los caracteres en mayúsculas son sustituidos por minúsculas.

```
$ more ejemplo5
mkdir Myfiles
cd Myfiles
ls -al MyFiles > AllofMyFiles
```

Con el comando

```
$ tr 'A-Z' 'a-z' < ejemplo5
```

tendríamos la salida que buscamos.

Viendo el comienzo de un fichero

El comando head nos permite ver el comienzo de un fichero. Por defecto nos muestra las 10 primeras líneas. Las opciones del comando head están en la tabla 2-7

Tabla 2-7 : Opciones utilizadas con head

Opción	Uso
-c NUMERO	Especifica el número de bytes a ser mostrados.
-n NUMERO	Muestra el NUMERO de líneas especificado.
-q	No muestra las cabeceras.
-v	Muestra las cabeceras.
--help	Muestra la ayuda
--version	Muestra la versión.

Esto es un ejemplo del comando head. La opción -v es usada para mostrar el nombre de fichero localizado en la cabecera.

```
$ head -n 3 -v namelist
```

Viendo el final de un fichero

Hay un comando que nos permite ver el final de un fichero. Como head, el comando tail muestra las últimas 10 lineas de un fichero por defecto. Hay además muchas opciones en el comando tail que se describen en la tabla 2-8

Tabla 2-8 : Opciones utilizadas con tr

Opción	Uso
-NUMERO	Especifica el número de líneas a ser mostrado.
+NUMERO	Especifica el número de líneas desde el comienzo a partir de donde empieza a mostrar.
--retry	Indica a las instrucción que se mantenga intentando abrir un fichero cuando este esté inaccesible.
-c NUMERO	Especifica el número de bytes a ser mostrados.
-f	Muestra las líneas y va mostrando líneas según se escriben en el fichero. Este comando puede venir bien para ficheros que crecen como por ejemplo los ficheros LOG.
-n NUMERO	Muestra el NUMERO de líneas especificado.
-q	No muestra las cabeceras.
-v	Muestra las cabeceras.
--help	Muestra la ayuda
--version	Muestra la versión.

Esto es un ejemplo del comando tail. Vamos a mostrar los 50 últimos bytes del fichero namelist.

```
$ tail -c 50 namelist
```

Juntando múltiples ficheros

El comando join en realidad busca en dos ficheros entradas comunes. Las entradas encontradas en los dos ficheros son mostradas en la salida estandar donde pueden ser redireccionadas a un fichero. Puedes combinar ficheros usando campos. El comando join usa campos de unión para combinar líneas de múltiples ficheros. Antes de usar el comando join, el fichero debe de comenzar con los campos de unión. Esto se consigue muchas veces con el comando sort basado en los campos que van a ser juntados. Así, si tu estás utilizando dos ficheros que contienen el nombre y el primer apellido y quieres juntar los ficheros utilizando el apellido, entonces los dos ficheros deben ser ordenados previamente utilizando el campo apellido.

El correcto funcionamiento del comando join es el siguiente:

```
$ join -options FILE1 FILE2
```

Algunas opciones del comando join:

Tabla 2-9 : Opciones utilizadas con join

Opción	**Uso**
-I	Especifica que caso es ignorado cuando se combinan los ficheros.
-1 FIELD	Especifica el campo en el fichero 1
-2 FIELD	Especifica el campo del fichero 2
-t char	Especifica el carácter separador del fichero de entrada y de salida.
-v FILE#	Se imprime una línea por cada línea no “pareada” encontrada en el fichero FILE#
--help	Muestra la ayuda
--version	Muestra la versión.

El siguiente es un ejemplo del comando:

```
$ join prueba1 prueba2
```

Dividiendo ficheros en varias partes

La utilidad SPLIT se usa para dividir un fichero largo en varios ficheros distintos. Esta utilidad crea ficheros de una cierta longitud, cuyo valor por defecto es de 1000 líneas, y los nombra de forma secuencial. Los nombres de los archivos están formados por un prefijo, de valor “x” por defecto, seguido por una combinación de letras que sigue el patrón de “aa”, “ab”, “ac”, etc. Si se deben crear más de 676 ficheros, la sintaxis será “zaa”, “zab”, etc. Cuando no se especifica ningún

fichero de entrada para la utilidad SPLIT, la entrada de datos estándar se utilizará por defecto.

La sintaxis correcta para la utilidad SPLIT es la siguiente:

```
$ split opciones FICHERO  PREFIJO
```

Hay varias opciones, mostradas en la tabla 2-10, que pueden ser utilizadas para personalizar la salida de la utilidad SPLIT.

Tabla 2-10 : Opciones utilizadas con SPLIT

Opción	**Uso**
-l LÍNEAS	Especifica el número de líneas de cada uno de los ficheros de salida.
-b BYTES	Especifica el número de bytes de cada uno de los ficheros de salida. Si el número está seguido por "b", la cantidad indicada será multiplicada por 512; si se utiliza una "k", la cantidad será multiplicada por 1024; y si se utiliza una "m", la cantidad será multiplicada por 1048576.
-C BYTES	Funciona de manera similar a la opción "-b". Mediante esta opción, se irán añadiendo líneas completas a cada uno de los ficheros de salida, sin superar el número de bytes indicado.
--verbose	Escribe un diagnóstico en el error estándar antes de que se abra cada uno de los ficheros.
--help	Muestra información de ayuda.
--version	Muestra información sobre la versión.

En el ejemplo siguiente, el fichero nameslist se divide en varios archivos cada uno de los cuales contiene cinco líneas.

```
$ split -l5 nameslist names
```

Listamos el contenido del directorio para ver los archivos nuevos

```
$ ls names*
```

Eliminando líneas repetidas de un fichero

Para eliminar las líneas repetidas y consecutivas de un fichero usamos el comando uniq, de modo que si tenemos en un fichero "a" las líneas:

1
3
4
4
3

Al ejecutar el comando

```
$uniq a
```

El resultado será:

```
1
3
4
3
```

ya que la única línea que se repite es la 3ª y la 4ª que tienen el número 4.

Otras opciones de este comando son:

Opción	**Función**
-c	Enumera el número de ocurrencias (líneas que se repiten)
-d	Solo conserva las líneas que se repiten
-u	Solo conserva las líneas que son únicas

Mostrando ficheros en otros formatos

Hay ocasiones en las que puede ser necesario mostrar ficheros que estén en formato NO-texto y para ello, puede usarse la utilidad OD. Cada una de las líneas de salida consiste en el OFFSET, que es el número de bytes desde el inicio del fichero, y los de grupos de datos del fichero de entrada que se muestran a continuación. Por defecto, OD escribe el OFFSET en octal, y cada uno de los grupos de datos del fichero son dos bytes que se escriben como un único número octal. Las opciones para esta utilidad se muestran en la tabla 2-11.

Tabla 2-11 Opciones utilizadas con OD

Opción	**Uso**
-A RADIX	Especifica la base utilizada para mostrar el offset. El valor del parámetro puede ser cualquiera de los siguientes: d se utiliza para decimal o se utiliza para octal x se utiliza para hexadecimal n se utiliza para especificar ninguna
-j BYTES	Especifica el número de bytes de entrada que deben ser descartados antes de empezar a mostrar datos.
-N BYTES	Especifica el número máximo de bytes a mostrar
-s [N]	Muestra cadenas de al menos N caracteres.
-t TIPO	Selecciona el formato en el que se mostrará la información de salida. TIPO es una cadena de caracteres formada por uno o más de los siguientes indicadores de caracteres: a para caracteres c para caracteres ASCII o secuencia de escape d para decimal con signo f para coma flotante o para octal u para decimal sin signo x para hexadecimal
-w BYTES	Estructura la salida con el número de bytes por línea indicado. El valor por defecto es 32 bytes.
--help	Muestra información y termina.
--version	Muestra información sobre la versión y termina.

A continuación se muestra un ejemplo del uso de la utilidad OD. En este ejemplo, el fichero "prueba1" se muestra con el offset en formato hexadecimal y los datos en octal, con una amplitud de ocho bytes.

```
$ od -A x -w8 prueba1
```

Consejo para el examen: Se debe prestar atención a las utilidades poco utilizadas como OD o JOIN, ya que son un excelente material para el examen.

Convirtiendo ficheros para imprimir

La utilidad PR formatea y prepara ficheros para imprimir, escribiéndolos en la salida estándar, paginándolos y opcionalmente escribiéndolos en un formato de multicolumna. Adicionalmente, también puede unir ficheros, imprimiéndolos en paralelo, uno por columna. La sintaxis para esta utilidad es la siguiente:

```
pr -opciones FICHERO
```

Por defecto, en cada página se escribe un encabezado de cinco líneas: dos líneas en blanco, una línea con la fecha, el nombre del archivo y el contador de página, y dos líneas más en blanco. Igualmente, también se escribe un pie de página de cinco líneas. Hay numerosas opciones para especificar el formato producido con la utilidad PR, algunas de las cuales se muestran en la tabla 2-12.

Tabla 2-12 : Opciones utilizadas con PR

Opción	**Uso**
-COLUMNAS	Produce tantas columnas como el número COLUMNAS y equilibra el número de líneas en cada columna dentro de cada página.
-a	Imprime las columnas en horizontal en lugar de en vertical.
-d	Inserta un doble espacio en la salida
-f	Utiliza saltos de página en lugar de caracteres de nueva línea para separar páginas.
-h TEXTO	Utiliza el texto especificado en TEXTO en lugar del nombre del fichero dentro del encabezado.
-l LÍNEAS	Establece el número de líneas por página
-m	Imprime todos los ficheros en paralelo, uno por columna.
-N NÚMERO	Empieza contando por NÚMERO en la primera línea de la primera página impresa.
-w CARACTERES	Establece el ancho de página a un número de caracteres igual a CARACTERES (el valor por defecto es 72). Solamente se utiliza para formatos de salida multicolumna.
-W CARACTERES	Establece el ancho de página a un número de caracteres igual a CARACTERES siempre. El valor por defecto es 72.

Mostrando ficheros al revés

La utilidad TAC se utiliza para mostrar líneas de un fichero al revés donde, por defecto, una nueva línea empieza después del carácter de salto de línea. Éste es el opuesto al comando CAT, tanto en el nombre como en su funcionamiento.

La sintaxis para la utilidad TAC es la siguiente:

```
$ tac – opciones FICHERO
```

Las opciones disponibles se muestran en la tabla 2-13.

Tabla 2-13 : Opciones utilizadas con TAC

Opción	**Uso**
-b	Adjunta el separador al principio de la línea que le precede en el fichero, en lugar de hacerlo al final.
-r	Trata la cadena de caracteres del separador como una expresión regular.
-s SEPARADOR	Utiliza el carácter SEPARADOR en lugar del carácter de salto de línea como el separador de registros.

Mostrando estadísticas de un fichero

La utilidad WC cuenta el número de bytes, palabras separadas por espacios en blanco y saltos de línea para cada uno de los ficheros indicados. Se muestra una línea de resultados para cada uno de los ficheros, y si el fichero fue indicado como un argumento, muestra su nombre a continuación. Si se indica más de un fichero, la utilidad muestra una línea final indicando los resultados acumulativos con el texto “total”.

El orden en el que se muestran los resultados es el siguiente: en primer lugar los saltos de línea, luego las palabras y finalmente los bytes. Por defecto, cada resultado se muestra justificado a la derecha en un campo de siete bytes con un espacio en blanco entre cada uno de los resultados, de manera que los números y los nombres de los ficheros se alinean correctamente en columnas. Algunas de las opciones disponibles con WC se describen en la tabla 2-14.

Tabla 2-14 : Opciones utilizadas con WC

Opción	**Uso**
-c	Muestra únicamente el número de bytes
-w	Muestra únicamente el número de palabras
-l	Muestra únicamente el número de líneas
-L	Muestra la longitud de la línea más larga
--help	Muestra información de ayuda y termina
--version	Muestra información sobre la versión y termina

A continuación se muestran algunos ejemplos:

```
$ wc prueba1
```

```
$ wc –c prueba1
```

Añadiendo números de línea a un fichero

La utilidad nl es útil para mostrar los números de línea de un fichero. Se organiza el fichero de entrada en páginas lógicas y por defecto, el número de línea se inicializa a 1 al principio de cada una de ellas. Se tratan todos los ficheros de entrada como un único documento y no se inicializan los números de línea ni las páginas lógicas entre ficheros.

Una página lógica consiste en tres secciones separadas por una línea en blanco: encabezado, cuerpo y pie de página. Cualquier de ellas puede estar vacía y puede estar numerada de una forma distinta a las otras dos. El texto que preceda el primer separador de sección en el fichero de entrada se considerará parte del cuerpo, de manera que la utilidad nl tratará un fichero sin delimitadores de sección como una única sección de cuerpo.

Numerosas opciones pueden utilizarse para personalizar el resultado generado con esta utilidad, y algunas se detallan en la tabla 2-15.

Tabla 2-15 ; Opciones utilizadas con nl

Opción	**Uso**
-a	Numera todas las líneas
-t	Numera únicamente las líneas no vacías
-n	No numera las líneas. Es el valor por defecto de los encabezados y los pies de página.
-i NÚMERO	Incrementa el número de línea en una cantidad igual a NÚMERO. El valor por defecto es uno.
-p	No inicializa los números de línea al principio de cada página lógica.
-s CADENA	Añade la cadena de caracteres CADENA después de cada número de línea.
-v NÚMERO	Establece el NÚMERO inicial de cada página lógica.
-w NÚMERO	Especifica el NÚMERO de espacios que se reservan para los números de línea. El valor por defecto es seis.

A continuación se adjunta un ejemplo del uso de la utilidad nl, en el que se muestran los números correspondientes a todas las líneas.

```
$ nl prueba1
```

Usando el editor de flujo (sed)

La utilidad sed es un editor de flujo que puede tomar su entrada de un fichero o de datos que le son pasados desde una tubería (pipe). El sed normalmente opera globalmente sobre un fichero a no ser de que se le haya especificado lo contrario con símbolos de direccionamiento (adressing), en cuyo caso limitan el ámbito de ejecución del comando. Que el sed opera normalmente de forma global quiere decir que cada vez que encuentra el patrón especificado, lo reemplaza. El direccionamiento puede ser usado para especificar el lugar donde se debe buscar para encontrar el patrón sobre el que actuar. Este direccionamiento puede especificar una sola línea o un grupo de

ellas. Así mismo se puede excluir la actuación del comando sobre determinadas lineas usando el símbolo de exclamación invertido (!). Las expresiones regulares también pueden ser usadas para especificar lugares dentro de un archivo.

El sed puede ser usado para realizar simples substituciones o cambios mucho más complejos y potentes en un fichero. Las substituciones simples sobre un archivo se realizan usando la siguiente sintaxis:

```
sed -opción s/expresión_regular/substitución/flag nombre.archivo
```

El sed también funcionara con texto pasado desde la entrada standard y con texto de archivos especificados. En este caso, el archivo original quedará intacto y los cambios serán escritos en uno nuevo. Las "expresiones regulares" son una forma de buscar caracteres concretos y este punto se trata en la siguiente sección. En este caso el parámetro "s" le dice al sed que localice la expresión_regular que le hayamos dado y la borre, poniendo en su lugar lo que hayamos puesto como substitución.

Se pueden realizar múltiples substituciones usando el parámetro "-e". Por ejemplo:

```
$ sed -e 's/lisa/Lisa/' -e 's/nikki/Nikki' amigos.mios
$ sed -e 's/lisa/Lisa/' ; 's/nikki/Nikki' amigos.mios
```

Esta orden en cualquiera de sus dos sintaxis, buscaran la expresión regular "lisa" y la substituirían por "Lisa" y "nikki" por "Nikki" en el fichero amigos.mios .

También se puede utilizar el sed para ejecutarlo con una serie de ordenes y parámetros previamente definidos en un script, permitiendo automatizar y almacenar las opciones de uso mas frecuente y simplificar comandos demasiado largos. En este caso se debe de usar con la opción "-f" de la siguiente forma:

```
$ sed -f mi.script archivo.mio
```

Si se usa con un script, el script debe contener la sintaxis de los parámetros del sed con el mismo formato que usaríamos en un caso sin script: s/expresión-regular/substitución/flags

Las opciones de uso mas frecuente con sed son:

Opción	**Uso**
-V	Muestra información sobre la versión y sale.
-h	Muestra la información de ayuda del comando y sale.
-n	Evita que se envíe por pantalla el fichero modificado.
-e comando	Añade el comando a las ordenes a realizar.
-f script	Añade las ordenes de un script a las que se tienen que realizar.

Los flags se pueden añadir cuando se usa el comando "s" (ejemplos anteriores de substitución) de manera que permiten refinar y mejorar sus acciones. Estos son los flags que se pueden usar.

Flag	**Uso**
g	Aplica los cambios de forma global
p	Imprime por pantalla solo las líneas afectadas por algún cambio
Número	Reemplaza solo el caso del número (ordinal) expresado
w archivo	Crea un archivo solo conteniendo las líneas modificadas
I	Ignora las mayúsculas al realizar la búsqueda de la expresión

Un ejemplo del uso de de sed con un flag numérico sería el siguiente:

```
$ sed  's/hola/HOOOLA/5' archivo.mio
```

En este caso solo el 5º "hola" que encontrase en archivo.mio, sería reemplazado por "HOOOLA" y daría toda la salida por pantalla.

Así mismo, sed puede usar direcciones para expresar el lugar de los cambios en un fichero. Estas direcciones pueden ser números de línea, símbolos de numero de línea, o incluso expresiones regulares. Estas expresiones para la localización las vamos a ver un poco mas abajo de forma mas extensa.

Estas expresiones de localización preceden a los comandos y parámetros pasados a sed.

Un ejemplo sería este:

```
$ sed '4.11s/hombre/Varon/' archivo.mio
```

En este caso, solo se sustituiría la expresión "hombre" por "Varon" de las lineas 4ª a 11ª (ambas inclusive) en el fichero archivo.mio, y daría la salida por pantalla de todo el fichero y sus cambios.

Así mismo, podemos indicar que ignore determinadas líneas al hacer la substitución. Este caso le indicaría que NO la realizase en la línea 6ª del fichero archivo.mio

```
$ sed '!6s/hombre/Varon/' archivo.mio
```

Este tipo de direccionamiento y localización de líneas se usa normalmente con estos parámetros:

Parámetro	**Uso**
NUMERO	Solo realizara la substitución sobre la línea NUMERO
NUMERO.NUMERO	Realizara las substituciones en las líneas comprendidas entre los dos números ambos inclusive
$	Especifica que solo se realice sobre la última línea
!	Se realizará sobre todas las líneas excepto las expresadas tras !

Ejercicios

Las siguientes preguntas y ejercicios te permitirán repasar los contenidos de este capítulo. Tómate tu tiempo para completarlas y repasa cualquier duda que tengas. El contestar la respuesta correcta no es tan importante como el entender correctamente la pregunta, así que si te es necesario, echa mano de cualquier material que te permita saber exactamente que te están preguntando.

Sentirte cómodo con las preguntas y respuestas de esta sección te ayudara a estar más preparado para el examen de certificación LPI.

Preguntas Pre-TEST

1. ¿Qué herramienta se usa para añadir el número de línea a un fichero ?
2. ¿Qué herramienta permite combinar dos ficheros usando campos de unión?
3. ¿Con qué utilidad se crean líneas de una determinada longitud en un fichero?
4. ¿Qué herramienta se usa para ver el contenido de un fichero a la inversa?
5. ¿Qué herramienta permite borrar caracteres de un fichero?
6. ¿Con qué herramienta se puede ver el contenido en hexadecimal de un fichero?
7. ¿Con qué utilidad se pueden ordenar alfabéticamente los contenidos de un fichero?

Preguntas TEST

1. ¿Qué comando buscará de la linea 2 a la 20 en el archivo records los caracteres 1st y los reemplazara por los caracteres first?

A. sed s2-20/1st/first/ records
B. sed 2-20s/1st/first/ records
C. sed s2,20/1st/first/ records
D. sed 2,20s/1st/first/ records

2. ¿Qué comando dividirá el archivo researchpaper en varios archivos que contengan cada uno de ellos 60 líneas? (Escoge todas las respuestas correctas)

A. split -60 researchpaper
B. split -C 60b researchpaper
C. split -C 60 researchpaper
D. split -l 60 researchpaper

3. ¿Qué utilidad se usa para combinar las líneas de dos archivos diferentes? (Escoge todas las respuestas correctas)

A. split
B. join
C. paste
D. cut

4. ¿Qué usarías para ver las 5 últimas líneas del archivo myfiles?

A. tac myfiles
B. tail myfiles
C. tac -5 myfiles
D. tail -5 myfiles

5. ¿Cuál de las siguientes respuestas te permitirían ver el archivo myfiles en formato octal? (Escoge todas las correctas).

A. od myfiles
B. od -t o myfiles
C. od -t x myfiles
D. od -o myfiles

6. ¿Qué respuesta ordenaría alfabéticamente el fichero mylist, numeraría las líneas y finalmente lo separaría en ficheros que contuvieran 60 líneas cada uno?

A. sort mylist | nl > -60 lists
B. sort mylist > nl > split -60 > lists
C. sort mylist | nl | split -60 lists
D. sort mylist | nl | tee lists | split -60 lists

7. La utilidad _____ sirve para ver el contenido de un fichero de forma inversa. (Rellena)

8. ¿Qué utilidad nos muestra el total de líneas de una archivo?

A. nl
B. ln
C. wc
D. tr

9. La utilidad _____ se usa para asegurarse que los archivos se verán igual, sin importar el sistema que se use para visionarlos, convirtiendo las tabulaciones en espacios.

10. ¿Qué utilidad intenta convertir las líneas de un archivo en líneas de la misma longitud?

A. nl
B. ln
C. fmt
D. expand

11. ¿Con qué utilidad convertiríamos todas las minúsculas de un archivo a mayúsculas?

A. cut
B. sed
C. tac
D. tr

12. ¿Cuál de las siguientes respuestas usarías para verificar que las líneas de un archivo están ordenadas alfabéticamente?

A. sort -c
B. sort -d
C. sort -v
D. sort -m

Escenarios

1. Necesitas imprimir el archivo grande que se ha creado en el ejercicio anterior de forma que cada pagina esté numerada y tenga una cabecera que contenga el nombre de tu empresa y la fecha actual. ¿Como realizarías esta tarea?

Respuestas PRE-TEST

1. La utilidad nl se usa para numerar las líneas de un archivo.
2. La utilidad join se usa para juntar líneas de diferentes archivos.
3. El comando fmt se usa para dar formato con lineas de igual longitud a un archivo.
4. El comando tac muestra por pantalla un archivo de forma inversa, lo contrario que cat.
5. Las utilidades tr y sed son usadas para borrar y substituir caracteres en un archivo.
6. La utilidad od se usa para ver archivos en forma octal, hexadecimal, y otros formatos.
7. La utilidad sort se usa para ordenar el contenido de un archivo en orden numérico o alfabético.

Respuestas TEST

1. D. Cuando se usa direccionamiento con la utilidad sed, los números de línea y los rangos buscados se especifican antes del comando "s". El rango va separado por una coma.
2. A y D. La opción -l se usa para especificar el numero de líneas que contendrá cada archivo tras el uso de split. En cualquier caso, cuando no se le da ninguna opción, el numero indicado se asume como el numero de lineas.
3. B y C. Las utilidades paste y join se usan para combinar lineas de un archivo. El comando split se usa para dividir un archivo en trozos. La utilidad cut borra texto de un archivo.
4. D. El comando tail se usa para ver el final de un archivo y la opción -5 indica el numero de lineas que se quiere ver.
5. A y B. La utilidad od muestra los ficheros en octal por defecto. La opción -t seguida de o, se usa para indicar el formato octal.
6. D. Las tuberías se usan para enviar datos de un comando a otro, por lo tanto A y B son incorrectas. La respuesta C no crea dos archivos, así pues solo la D es la respuesta correcta.
7. tac. La utilidad tac se usa para ver un fichero comenzando desde la ultima linea hasta la primera.
8. C. La utilidad wc se usa para conocer los totales de un archivo, bien sean líneas, palabras o bytes.
9. expand. La utilidad expand se usa para convertir las tabulaciones (el carácter "tab") en espacios.
10. C. Con fmt se intenta crear líneas de igual tamaño en un fichero.
11. D. Usamos tr para borrar o substituir caracteres en un archivo.
12. A. Hay que usar la opción -c con sort para verificar que el contenido del fichero se ha ordenado.

Respuestas Escenarios

1. pr -h "Nombredelaempresa" largefile.

El comando pr automáticamente imprime la fecha y el numero de pagina como parte de la información de cabecera. Para añadir el nombre de la empresa la opción -h es la correcta, dándole lo que se desea incluir entre dobles comillas. El fichero, largefile, es el nombre del que creamos en el escenario anterior, que ahora imprimimos con el comando pr.

Bibliografia y enlaces recomendados

LPIC 1 Certification Bible (Bible) by Angie Nash, Jason Nash
John Wiley & Sons; Bk&CD-Rom edition (July 1, 2001) ISBN: 0764547720

LPI Linux Certification in a Nutshell by Jeffrey Dean
O'Reilly & Associates; 1st ed edition (May 15, 2001) ISBN: 1565927486

CramSession's LPI General Linux Part 1 : Certification Study Guide
CramSession.com; ISBN: B000079Y0V; (August 17, 2000)

Referencias Unix Reviews
http://www.unixreview.com/documents/s=7459/uni1038932969999/

Página LPI: www.lpi.org

Apuntes IBM: http://www-106.ibm.com/developerworks/edu/l-dw-linux-lpir21-i.html

Manuales GPL: http://www.nongnu.org/lpi-manuals/