Preparación para el examen LPI 101

Tema 103.8

Usando el editor vi

Créditos y licencia de uso

Coordinación:
Manuel Guillán (xLekOx) lpi@xlekox.org
Kiefer Von Jammo (Kiefer) kiefer@khrooon.net

Traducción:
Carmen Eugenio (nemrac) meneiro@ono.com
Manuel Guillán (xLekOx) lpi@xlekox.org

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

Versión 1.0 (16-08-2004 16:00)

ÍNDICE

Índice de contenido

Tema 103.8

Introducción

En este capítulo se verá el funcionamiento básico de uno de los editores de texto plano más famosos en el mundillo GNU/Linux. Navegar, insertar, copiar, mover, pegar y buscar, serán algunas de las tareas que se verán en este capítulo, y que servirán de ahora en adelante para editar los archivos de texto, ya sean textos o bien ficheros de configuración del sistema.

El único comando que se verá será el vi y algunas de sus funciones:

/, ?
h, j, k , l
G, H, L
i, c, d, dd, p, o, a
ZZ, :w!, :q!, :e!

Este tema tiene un peso (importancia) de 1 de cara al examen final de la certificación LPI 101. El total de la suma de pesos de todos los temas es de 106. A pesar de ello se recomienda el conocimiento de este potente editor.

Abriendo ficheros para su edición

Para abrir un fichero y así poder editarlo con vi, se usa el comando siguiente:

```
$vi  nombrefichero
```

Si no se especifica un nombre de fichero, vi abre uno en blanco, y luego se puede especificar el nombre al guardarlo más tarde. Hay que tener en cuenta los permisos de lectura/escritura del fichero, o no se podrán guardar los cambios.

Si se sabe la línea que se necesita editar, se puede ir directamente a esa línea usando la siguiente sintaxis:

```
$vi +10 nombrefichero
```

Con esto se le indica a vi que empiece editando el fichero en la línea 10. Otra opción para saltar rápidamente a un punto del fichero es especificar una cadena de búsqueda por donde empezar. Sería con la siguiente sintaxis:

```
$vi +/telnet inetd.conf
```

Esto arrancaría vi, abriría el fichero inetd.conf, y luego iría a la primera palabra que coincida con telnet.

vi tiene tres modos de actuar:

Modo comando
Modo insertar
Modo última línea

Cuando arranca vi, el modo por defecto es por comando. En modo comando se puede mover el cursor por todo el documento, poner comandos, borrar y añadir texto, o cambiar a modo insertar. Para volver al modo comando hay que pulsar la tecla Esc. Cuando se está en modo insertar, solo se puede insertar texto y usar la tecla suprimir para borrar los errores. No permite en este modo el mover el cursor por el documento. Para ello hay que volver al modo comando. El modo última línea se usa para entrar comandos complejos. Para entrar en ese modo se escriben desde modo comando dos puntos (:). Después de completar el comando se vuelve al modo comando.

Recordar que en vi los comandos son sensibles a las mayúsculas/minúsculas. J es diferente de j.

Saliendo de vi y guardando los archivos

Para salir de vi se hace desde el modo última línea, de manera que el comando debe empezar con dos puntos desde el modo comando.

Guardar sin salir.
Para guardar el fichero, se usa el comando :w- Para evitar mayores problemas se debería hacer esto con frecuencia.

Guardar y salir.
Para salir de vi guardando los cambios, se puede usar tanto el comando :wq como simplemente ZZ-

Salir sin guardar.
Si se necesita rehacer los cambios que se han hecho a un fichero, se debe salir de vi sin guardar los cambios. Este se puede hacer con el comando :q! La exclamación anula la necesidad de guardar el fichero antes de poder salir del programa.

Moviendo el cursor

Vi es un editor en modo texto, por lo que no está disponible el soporte del ratón. Esto suele a veces desalentar a los nuevos usuarios, superándose rápidamente con un poco de práctica. Vi ofrece numerosos atajos para moverse rápidamente por los ficheros, de manera que los usuarios no se atranquen con los simples movimientos del cursor.

La mayoría de las veces se pueden usar las teclas de flechas para moverse por el documento. Pero no todos los sistemas tienen estas teclas, y a veces las incompatibilidades de la emulación de un terminal causa que no funcionen como se espera. Las teclas para mover el cursor se muestran a continuación, colocadas según su función:

```
        k
   h         l
        j
```

La tecla - (menos) se puede sustituir por k, y la tecla + (más) por la j. Se deben aprender estas teclas incluso aunque se use siempre las flechas. Muchas otras aplicaciones usan las teclas de movimiento del cursor de vi dado que son populares y familiares. Por si solas, mueven el cursor una línea o carácter. Para mover más, se puede usar un número, como 4j para moverse 4 líneas hacia abajo.

Para moverse por el fichero más rápidamente, se puede usar los comandos de página completa o media página. Estos son Ctrl+f y Ctrl+b para moverse hacia delante o hacia detrás una página, y Ctrl+d y Ctrl+u para moverse hacia delante o hacia atrás media página.

También se pueden usar los números de línea para moverse por un fichero. Para ver los números de las líneas se usa el comando Ctrl+g.. Para moverse a una línea específica se usa el comando nG, donde n es el número de línea.

Hay muchos otros comandos para mover el cursor. La mayoría de ellos se muestran en la siguiente tabla. Recordar que se puede hacer que un comando se repita poniéndole un número delante.

Tabla 8-1 Comandos para mover el cursor

Tecla	***Función***
h	Mueve el cursor un carácter hacia la izquierda
j	Mueve el cursor hacia abajo una línea
k	Mueve el cursor hacia arriba una línea
l	Mueve el cursor un carácter hacia la derecha
Ctrl-G	Muestra el número de la línea en la que se encuentra
nG	Va a la línea especificada en n
Ctrl-f	Avanza una pantalla
Ctrl-b	Retrocede una pantalla
Ctrl-d	Avanza media pantalla
Ctrl-u	Retrocede media pantalla
Ctrl-e	Se mueve una línea hacia arriba
Ctrl-y	Se mueve una línea hacia abajo
w	Va al principio de la siguiente palabra
e	Va al final de la siguiente palabra
E	Va al final de la siguiente palabra. A diferencia del comando e, E considera la puntuación como parte de la palabra.
b	Va al principio de la palabra anterior.
0	Va al principio de la línea
^	Va a la primera palabra de la línea en la que se encuentra.
$	Va al final de la línea
Enter	Va a la siguiente línea.
-	Va al principio de la línea anterior.
G	Va al final del fichero.
%	Va a la agrupación que coincida
H	Va a la primera línea de la pantalla.
M	Va a la mitad de la pantalla.
L	Va al final de la pantalla.
n\	Mueve el cursor a la columna n.

Guardando una parte del fichero

No solo se puede salvar el fichero entero, vi permite guardar parte de un fichero especificando el número de líneas que se quiere escribir. Se puede uno figurar que cada línea se puede referenciar por un número, y esta operación se haría con la siguiente sintaxis:

: primera_línea , ultima_línea w nombreFichero

Especial atención a la w que va detrás de la última línea, que se usa para identificar la operación como de escritura. Hay dos caracteres que se pueden usar para especificar el número de línea:

$ significa la última línea del fichero
. significa la línea actual

Tabla 8-2 Algunos ejemplos de comandos para guardar parte de un fichero:

Secuencia	***Resultado***
:., 12w ficheroNuevo	Guarda la línea desde donde está el cursor hasta la línea 12 en un fichero llamado ficheroNuevo
:2, 5w ficheroNuevo	Guarda las líneas 2 a 5 en un fichero llamado ficheroNuevo.
:12, $w ficheroNuevo	Guarda las líneas desde la 12 hasta el final del fichero en un fichero llamado ficheroNuevo

Añadiendo texto

Se puede añadir texto de diferentes maneras en vi. Cualquiera de ellas precisa que el usuario esté en modo comando.

Insertando texto

Para entrar en modo insertar, usar el comando i. Esto hace que el texto se inserte en el documento dondequiera que esté el cursor. Para volver al modo comando, usar la tecla Esc.

Añadiendo texto

Para añadir texto desde la localización del cursor, usar el comando a. Para añadirlo al final de la línea actual, usar el comando A.

Creando una línea nueva

Para empezar una nueva línea de texto se pueden usar dos comandos. El primero, o, abre una nueva línea de texto por debajo del cursor. El segundo, O, crea una nueva línea de texto por debajo de la localización del cursor.

Cambiando texto

Vi usa un comando doble para cambiar el texto. Cuando se usa, vi marca las secciones a ser cambiadas. Después de marcarlo, el usuario entra el nuevo texto para reemplazar el viejo.

Comienza por el comando c. La segunda parte del comando le dice a vi que texto se ha de cambiar. Para cambiar una palabra, se usa cw, y para cambiar una frase entera cs. Vi usa un espacio para final de palabra y un punto para final de frase. Para cambiar el texto desde la posición actual al final de la línea, usar c$ o C.

Como muchos otros comandos, se puede usar la tecla para repetirlo. Por ejemplo, para cambiar tres palabras se pondría 3cw.

Sustituyendo texto

Para sustituir el texto existente no siempre necesitas contar el número de palabras o frases a cambiar. El comando r permite reemplazar un simple carácter en la posición que esté el cursor, y el comando R permite remplazar todo el texto hasta que se vuelve al modo comando con Esc.

Tabla 8-3 Lista completa de comandos para añadir texto

Comando	*Función*
i	Inserta texto a la izquierda del cursor.
I	Inserta texto antes del primer caracter en la línea que no es un espacio.
a	Añade texto a la derecha del cursor.
A	Añade texto al final de la línea en la que está.
o	Empieza una nueva línea de texto por debajo de la línea actual.
O	Empieza una nueva línea de texto por encima de la línea actual.
cw	Cambia una palabra.
cs	Cambia la frase actual.
C$ o C	Cambia la línea actual.
r	Reemplaza un solo carácter.
R	Reemplaza texto hasta que se pulsa la tecla Esc.
s	Sustituye texto por el carácter actual.

Borrando texto

Como se puede observar, vi ofrece muchas maneras diferentes de hacer las cosas, y borrar textos no es una excepción.

Borrando caracteres

Para borrar un carácter de texto se usa los comandos x y X. El comando x borra el carácter seleccionado por el cursor, mientras que el comando X borra el carácter que precede al cursor.

La función repetir también se usa con estos comandos. Para borrar ocho caracteres antes del cursor, se usuaria el comando 8X.

Borrando palabras

El comando dw borra la palabra actualmente seleccionada. Para seleccionar una palabra basta con situar el cursor en la primera letra. Para borrar varias palabras a la derecha del cursor poner un número antes o después de la d, como en 3dw o d3w.

Borrando líneas

Para borrar la línea de texto en la que está situado el cursor, usar el comando dd. Para borrar múltiples líneas se añade un número, como por ejemplo 3dd para borrar tres líneas de texto.

Borrando texto desde el cursor.

Dos comandos se usan para borrar todo el texto antes o después del cursor. Para borrar todo el texto después del cursor hasta el final de línea, usar el comando D, y para borrar todo el texto desde el

principio de la línea hasta el cursor, usar d^. Para borrar todo el texto desde la posición actual del cursor hasta el final de la pantalla, usar dL, y para borrar todo el texto hasta el final del fichero, usar dG. Para borrar todo el texto desde el cursor hasta el principio del fichero, usar dlG. Para borrar todo el texto desde el cursor a una línea específica, usar el comando d#$, donde # es el número de la línea.

Tabla 8-4 resumen de los diversos comandos para borrar

Comando	***Función***
x	Borra el carácter donde se encuentra el cursor.
X	Borra el carácter anterior a donde se encuentra el cursor.
dw	Borra una palabra.
dd	Borra una línea.
D	Borra todo el texto después del cursor hasta el final de la línea.
dL	Borra todo el texto después del cursor hasta el final de la pantalla.
dG	Borra todo el texto desde el cursor hasta el final del fichero.
d^	Borra todo el texto desde el principio de la línea hasta el cursor.
dlG	Borra todo el texto desde el principio del fichero hasta el cursor.
d#$	Borra todo el texto desde el cursor hasta el número de línea especificado.
d$	Borra desde la situación actual del cursor hasta el final de línea
d)	Borra el resto de la frase.
d}	Borra el resto del párrafo.
d0	Borra desde la situación actual del cursor hasta el principio de la línea.
db	Borra la palabra anterior.
dl	Borra una letra.
7dl	Borra siete letras.
7dw	Borra siete palabras.

Copiando y pegando

Como otros editores de texto modernos, vi ofrece las opciones de copiar y pegar texto. Esto puede ser de gran ayuda para realizar con más rapidez trabajos repetitivos, editando textos como por ejemplo shell scripts.

Yanking and pasting text
Yanking y pasting son los términos usados por vi para copiar y pegar. Para copiar una línea de texto es necesario usar la combinación de teclas yy o Y, añadir un número al inicio de las combinaciones hará que se copie ese número de líneas, por ejemplo 5Y para copiar 5 líneas. Para pegar texto, se usa el comando p.

En vi se usan dos tipos de buffer:
* El predeterminado (unnamed buffer), cuando se copia una porción de texto, esta va a parar a este buffer, sobreescribiéndose el anterior texto que hubiera en él.
* Y el named buffers, estos al contrario que el predeterminado se les llama por un nombre (una

letra) y permiten más operaciones como veremos.

Usando yy o Y el texto se copia al buffer (unnamed buffer). Vi soporta hasta 26 buffers diferentes (named buffers). El unnamed buffer es útil para copias rápidas, pero se pueden perder por accidente, por eso es útil tener varios named buffers disponibles (estos no pierden su contenido si no se le indica). Para diferenciar entre un comando y un named buffer se usa el símbolo " (comillas). Los named buffers son representados por un solo carácter. Cuando se copia texto, las letras minúsculas se usa para sustituir el texto que está en el buffer, sin embargo las mayúsculas se usan para añadir texto al buffer. Por ejemplo, para mover la línea actual al buffer b, se usa:

"bY o "byy

Para copiar 3 líneas al buffer:

"by3

Y para añadir 3 líneas al buffer actual:

"By3

Este es un buen ejemplo de como los comandos pueden ser apilados en vi. Para copiar palabras en vez de líneas se usa el comando yw. Para mover 3 palabras al buffer c, se usaría: "Cy3w

Para pegar texto del buffer c: "Cp

Moviendo texto

Para mover texto de un lugar a otro del documento, se usa el comando dd. Por si mismo, el comando pone el texto en el unnamed buffer, pero también se puede añadir a un named buffer. Para borrar el texto y ponerlo en el buffer se usa el comando: "add

O para añadir 5 líneas: "a5dd

Para pegar el texto movido, se usan los mismos comandos que para pegar texto copiado.

Copiando y moviendo entre ficheros

El unnamed buffer puede ser usado solamente para un único fichero, sin embargo, los named buffers se pueden usar para mover texto entre ficheros. Para realizar esta operación, se pone el texto en un named buffer y se abre otro fichero con el comando :e. Para abrir el fichero llamado lista.txt: :e lista.txt

Una vez el fichero está abierto, puedes pegar el texto como se haría normalmente. Por ejemplo, para mover 5 líneas del fichero original al fichero nuevo:
"a5Y
:e lista.txt
"ap
:w
:e lista_vieja.txt

Esto copia 5 líneas del fichero lista_vieja.txt, abre el fichero lista_nueva.txt, pega el contenido del buffer, salva el fichero y por ultimo vuelve a abrir el fichero original. La tabla 8-4 hace un resumen de los comandos de copiado y pegado.

Tabla 8-5 Comandos de copiado y pegado

Comando	***Uso***
yy or Y	Copia una línea en el unnamed buffer
nyy or nY	Copia n líneas al unnamed buffer
yw	Copia una palabra en el unnamed buffer
ynw or nyw	Copia n palabras en el unnamed buffer
y$	Copia el texto desde el cursor hasta final de línea en el unnamed buffer
"ayy	Copia una línea en el named buffer a
"Ayy	Añade una línea en el named buffer a
p	Pega el contenido del buffer a la derecha del cursor
P	Pega el contenido del buffer a la izquierda del cursor
np	Pega n copias del texto a la derecha del cursor
"ap	Pega el contenido del named buffer a a la derecha del cursor
"c3P	Pega 3 copias del buffer c, a la izquierda del cursor
"add	Mueve la línea actual al buffer a
"a5dd	Mueve 5 líneas al buffer a
dw	Borra una palabra y la pone en el unnamed buffer

Buscando texto

Además de las habilidades para manipular texto, vi también ofrece potentes mecanismos para buscar en los ficheros. Vi ofrece la posibilidad no solamente buscar simples palabras o frases como otros editores, sino también de buscar mediante expresiones regulares.

Buscando mediante patrones (patter matching)

Los caracteres especiales que usa vi con las búsquedas amplían las capacidades del editor de texto más allá de simples palabras o frases. Estos caracteres se les conoce como metacaracteres.

Buscando un carácter simple
Para buscar un carácter simple, se usa el . (punto). Por ejemplo, para buscar casas y casos:
cas.s

Buscando caracteres múltiples.
El * se usa para buscar caracteres múltiples, también buscará el carácter vacío. Por ejemplo, m*n buscará mano, montón, mon, y también mn.

Principio y final de línea .
Se puede indicar que un patrón coincida solamente si está al final o al principio de las líneas. El símbolo ^ define el principio de línea y el símbolo $ define el fin. Recordando que las búsquedas son sensibles a las mayúsculas-minúsculas (case-sensitive). Por ejemplo, el resultado de buscar la cadena "^El" (sin comillas) coincidirá con las dos primeras frases:

El perro se llama Bob.
Elias compró un coche.
el camión en rojo.

El resultado de la cadena "kernel$" coincidirá con la primera y tercera línea :

Ayer puse el nuevo kernel
A Maria no le gusta tocar en el kernel.
Mario tiene problemas con su kernel

Buscando un metacaracter
En algunos casos se necesita buscar un carácter que puede ser interpretado como un metacarater, como un punto o el símbolo del dólar, para evitar la sustitución y que vi interprete exactamente lo que se quiere buscar, se tiene que indicar cuando se trata de una situación como esta (a esto se le llama "escaping the character") se emplea la barra invertida (\ ) seguido del metacaracter, por ejemplo para buscar "[...] pedro. A la hora [...] Se usa:
pedro\. A la hora

Buscando un rango de caracteres
A veces puede resultar útil la búsqueda de caracteres que se encuentran en un rango dado, esto se consigue poniendo entre corchetes el rango deseado. Por ejemplo para buscar todas las entradas v2.x se usa:
v2\.[1-9]

Recordando que se necesita usar la barra invertida para buscar el . . Todos los números entre el 1 y el 9 se incluirán en el resultado. El carácter ^ se puede emplear para buscar lo contrario de lo que ponemos, por ejemplo para buscar cualquier letra que no sea dos: [^2]

En la tabla 8-6 se da un listado de los metacaracteres más comunes

Tabla 8-6 Metacaracteres más comunes

Metacaracter	***función***
\	Búsqueda de metacaracteres
.	Cualquier carácter
*	Ninguno o varios caracteres
[]	Rango de caracteres
^	Busca caracteres al principio de línea
$	Busca caracteres al final de línea
[^a]	Excluye letras de la búsqueda
[a-b]	Busca cualquier resultado que esté el rango

Buscando texto
vi ofrece 2 métodos para buscar en el texto. El primero es muy simple y busca solamente por un carácter. Para buscar la siguiente referencia del carácter a, se pone el comando fa. Para buscar una referencia de a antes del cursor se usa el comando Fa. Si no se quiere tener el cursor situado en el carácter buscado, se pueden usar los comandos T y t. Si se busca por ta, el cursor se situa a la izquierda de a. Si se usa Ta en el siguiente carácter a la derecha.

Para buscar más de un simple carácter, se usan los comandos / y ?. Cuando se está en el modo comando, se puede teclear el metacaracter / para buscar la primera referencia del texto a la derecha del cursor, para buscar a la izquierda se usa el comando ?, se pueden usar expresiones regulares cuando se está buscando con los metacaracteres / y ?

Repitiendo una búsqueda
Para repetir una búsqueda simple, se usa la tecla ; Para cambiar la dirección de la primera búsqueda simple, se usa el metacaracter , (coma).

Para repetir la búsqueda realizada con / o ?, se pulsa otra vez el mismo metacarácter (/ o ?) seguido de enter. Se puede elegir la misma búsqueda y cambiar de dirección al mismo tiempo. Un acceso directo para repetir la búsqueda en la misma dirección es usando el comando n, y para la dirección opuesta el comando N.

Buscando y reemplazando texto

Los mecanismos de vi ofrecen la posibilidad de buscar un texto y reemplazarlo. Estas funciones se pueden usar en determinados bloques de texto a lo largo del fichero. Estos comandos se deben usar desde el prompt colon. Para buscar y reemplazar el texto en el fichero entero, se utiliza el comando :% y para buscar desde el cursor en adelante el comando :.,$ Para buscar en un determinado rango de líneas se usa :1,5 el cual se aplicará desde la línea 1 hasta la 5. Para las siguientes líneas relativas desde la posición del cursor :/+3, lo cual referencia a la 3ª línea desde la posición actual. La sintaxis para la búsqueda y sustitución es:

:<inicio>,<finalizacion>s/<encontrar>/<reemplazar>

Por ejemplo, para reemplazar la palabra kernel por módulo en la primera instancia de cada línea , se usa

:%s/kernel/modulo

Para hacer lo mismo en todas las líneas se debe añadir la opción g tal como se muestra en el siguiente ejemplo:

:%s/kernel/modulo/g

Esto realiza una búsqueda y sustitución completa automáticamente. Para que el vi actúe de modo interactivo, es decir, que pida confirmación de cada cambio al usuario, se usa la opción c:
:%s/kernel/modulo/gc

La tabla 8-7 muestra los comandos más comunes de las búsquedas

Tabla 8-7 Comandos de búsqueda

Comando	***función***
fa	Busca la primera referencia de a a la derecha del cursor
Fa	Busca la primera referencia de a a la izquierda del cursor
ta	Se mueve al carácter a la izquierda de a
Ta	Se mueve al carácter a la derecha de a
;	Repite la búsqueda realizada por los comandos f,F,t o T
,	Misma función que ; pero en orden inverso
/string	Busca a la derecha la referencia del string
?string	Busca a la izquierda la referencia del string
n	Repite la búsqueda realizada con / o ?
N	Repite la búsqueda realizada con / o ? en orden inverso
:%	Busca en el fichero entero
:.,$	Busca desde la posición del cursor en adelante
:1,5	Busca en las líneas de la 1 a la 5

Deshaciendo cambios

Por norma general todo el mundo comete errores, y como no podía ser de otro modo, vi ofrece la posibilidad de deshacer los cambios que se han hecho en la línea actual. El comando undo (u) deshace la acción anterior (y sólo la acción anterior). Tiene un complemento -U- el cual deshace todas las acciones previas en la línea en que se encuentre. Es importante observar, sin embargo, que U solo almacena los cambios de una línea (la actual); esto impide que se hagan cambios a cuatro líneas y luego volviendo a la primera se intente deshacer todo. Si realmente se hizo un estropicio se puede volver atrás con la última copia salvada con el comando :e!

Otras operaciones

Con tantas operaciones que se han visto en este capítulo sobre los números de línea – se puede mover a una línea en particular especificando su número, se pueden guardar sólo unas líneas específicas, etc... es a menudo de gran ayuda ver las líneas numeradas. En vi se activa la numeración de líneas cambiando a modo comando con el símbolo de dos puntos (:) - también llamado modo dos puntos -, mover el cursor al final de la pantalla, entrar el comando set number y presionar Enter. Esto activa la numeración de líneas.

Los números sólo aparecen en el editor, como si los estuviéramos viendo con el comando nl, y no se guardan con el fichero. Set number se puede abreviar como set un (para deshacer el número, usar tanto set nonumber como set nonu). Si sólo se quiere visualizar el número de línea en la que se encuentra actualmente, se puede usar la combinación de teclas Ctrl+G. Los números de línea (así

como el número total de líneas dentro del fichero) aparecen al final de la pantalla. Según sea su ejecución, la combinación de teclas puede también mostrar otros items como el porcentaje de fichero antes del cursor, o la columna actual.

Si se desea ejecutar un comando externo dentro del editor, se puede usar la siguiente sintaxis:

:![comando]

Por ejemplo, para ver una lista de los archivos del directorio actual mientras se trabaja desde vi, el comando sería:

:!ls -l

Este comando muestra la lista y luego indica que se presione Enter para volver al fichero en el que se estaba trabajando.

Si se necesita copiar el contenido de otro fichero dentro del actual, se puede utilizar la siguiente sintaxis:

:r [nombreFichero]

Por ejemplo, para incluir el contenido de un fichero llamado *primero* dentro del fichero actual, el comando sería:

:r primero

Esto inserta el texto desde el otro fichero directamente en el actual en donde esta situado el cursor.

Preguntas Test

1. Se está trabajando con vi en un fichero que necesita modificaciones importantes. El cursor está en la primera letra de una frase que dura tres líneas. Que comando se puede usar para borrar toda la frase en cuestión?
   a. dd
   b. d$
   c. d)
   d. dw

2. Mary ha abierto un fichero log desde vi y necesita moverse a la última línea para ver la entrada más reciente. Que combinación de teclas debe introducir para conseguirlo?
   a. G
   b. X
   c. ZZ
   d. $

3. Cual de las siguientes combinaciones de teclas se pueden usar para salir del editor vi?
   a. :w
   b. :q
   c. :q!
   d. ZZ

4. Evan tiene el cursor en la mitad de una línea y necesita cambiar el fichero desde donde está el cursor hasta el final de la línea. Que secuencia debería usar para decirle al editor vi lo que quiere hacer?
   a. c^
   b. C
   c. .c}
   d. .r

5. Qué secuencia se usa para saber en que número de línea se encuentra situado el cursor?
   a. cw
   b. /
   c. .Ctrl+F
   d. . Ctrl+G

6. Estás editando un fichero y quieres buscar el número de empleado “12345“. Este número puede aparecer por muchos sitios del fichero, pero se refiere al número de empleado sólo si son los primeros 5 caracteres de la línea. Cómo indicamos la búsqueda correcta?

7. Estás editando un fichero y quieres buscar “home“. Este texto puede aparecer en muchos sitios, pero significa que es el empleado sólo si estos cuatro caracteres son los primeros dentro de la palabra. Que se debe introducir para iniciar una búsqueda correcta?

8. Qué secuencia se debe usar para copiar ocho líneas en el buffer para poder ser copiadas en otra localización dentro del fichero?
   a. 8cc
   b. cc8
   c. 8pp
   d. .pp8
   e. 8yy
   f. yy8

9. Estás al final de un fichero muy largo y quieres buscar hacia atrás la ciudad “Muncie“. Que secuencia usarías para buscar hacia atrás y encontrar la primera coincidencia con esta palabra?
   a. /Muncie
   b. :Muncie
   c. .Ctrl+F
   d. ?Muncie

10. En muchas versiones de Linux, que editor se incluye que arranca automáticamente -y transparentemente - al teclear vi?
    a. vim
    b. emacs
    c. de
    d. edlin

Respuestas Test

1. La respuesta correcta es c, que borrará toda la frase. La respuesta a borra la línea, mientras que la respuesta b borra desde el cursor hasta el final de la línea. La d solo borra la primera palabra.

2. La respuesta correcta es a porque el comando G sitúa el cursor al final del fichero. La respuesta b es incorrecta porque X borra el carácter antes del cursor. La respuesta c es incorrecta porque ZZ guarda el fichero y sale del editor. La respuesta d es incorrecta porque $ significa final de línea , pero no de fichero.

3. Las respuestas correctas a esta pregunta son b, c y d. La respuesta b vale, pero solo se puede usar si no se han hecho cambios al fichero. La respuesta c vale y puede usarse si se han hecho cambios pero no se desea guardarlos. La respuesta d guarda el fichero y sale del editor. La respuesta a es incorrecta porque guarda el fichero, pero no sale del editor.

4. La respuesta correcta es b, que indica que los cambios se han de hacer desde la situación actual del cursor hasta el final de la línea. La respuesta a es incorrecta porque indica que los cambios se hagan desde la localización actual hasta el principio de la línea. La respuesta c es incorrecta porque indica que se hagan los cambios en el resto del párrafo. La respuesta d es incorrecta porque permite cambiar sólo un carácter.

5. La respuesta correcta es d, que muestra la línea actual y otra información al final de la pantalla. La respuesta a es incorrecta porque es para cambiar una palabra. La respuesta b es incorrecta porque es para cambiar al modo de búsqueda. La respuesta c es incorrecta porque permite moverse por el fichero hasta la siguiente pantalla.

6. La respuesta correcta es "/^12345"
   El "/" se necesita para introducir cadenas de búsqueda, y el carácter " ^ " significa que la cadena debe aparecer al principio de la línea para ser relevante.

7. La respuesta correcta es " / \ < home"
   El " / " se necesita para introducir cadenas de búsqueda, y la secuencia " \ < " identifica que el texto debe estar al principio de la línea.

8. La respuesta correcta es e, que copia las siguientes ocho líneas en el buffer. La respuesta a es incorrecta porque permite cambiar (no copiar) las siguientes ocho líneas. La respuesta b es incorrecta porque usa una sintaxis incorrecta y no funciona. La respuesta c tampoco funcionaria porque no existe dicho comando – mismo problema en la respuesta d. La respuesta f falla porque solo extrae una línea. Haría "algo" 8 veces – dependiendo de lo que se escriba después del ocho.

9. La respuesta correcta es d, la cual busca hacia atrás por el fichero hasta que encuentra la primera coincidencia. La respuesta a busca hacia adelante desde la situación del cursor para encontrar la primera coincidencia. La respuesta b permite entrar un comando (los dos puntos), y recibe el comando desconocido de "Muncie" y devuelve un error. La respuesta c simplemente mueve el cursor a la siguiente pantalla y no realiza ninguna búsqueda.

10. La respuesta correcta es a, que es una versión mejorada de vi. La respuesta b es un editor que se puede bajar gratuitamente y usar en lugar de vi. La respuesta c es un viejo editor que existía en Unix y que era casi imposible trabajar con el. La respuesta d es un viejo editor de los primeros días del DOS y solo era ligeramente más fácil de usar.

Bibliografía 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/