U3 ADMINISTRACIÓN DE SOFTWARE BASE

ACTIVIDAD PRÁCTICA 7: Gestión de Procesos 2

Sergio Cobo García

SISTEMAS INFORMÁTICOS

Contenido

[Comunicación entre procesos y utilización de filtros 2](#_Toc125409210)

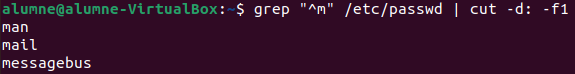
# Comunicación entre procesos y utilización de filtros

|  |  |
| --- | --- |
| 01 | Respecto al paginador **less**, indique cómo se pueden llevar a cabo las acciones indicadas en la mesa de abajo: |

|  |  |
| --- | --- |
| PAGINADOR LESS | |
| Avanzar página a página | ESPACIO o z |
| Retroceder una página | b o w |
| Avanzar línea a línea | ENTER, Flecha hacia abajo, j o e |
| Retroceder línea a línea | y, Flecha hacia arriba o k |
| Buscar una palabra, hacia adelante | /*palabra* (ejemplo: /termino) |
| Buscar una palabra, hacia atrás | ?*palabra* (ejemplo: ?termino) |
| Ir a una línea específica | g*NumeroLinea* (ejemplo: g100) |
| Salir | q, Q o ZZ |

|  |  |
| --- | --- |
| 02 | En Linux, escriba un comando que muestre **ÚNICAMENTE los nombres de los usuarios** que tienen un nombre que comienza por ‘m’. ***Pista****: Debe trabajar con el archivo /etc/passwd*. |

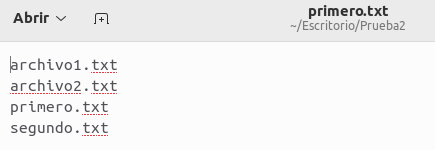
El comando sería **grep “^m” /etc/passwd | cut -d: -f1**.

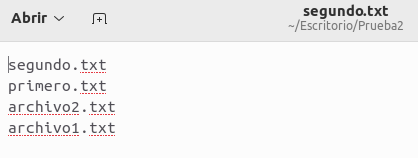


|  |  |
| --- | --- |
| 03 | En Linux, ¿Qué efecto tiene el comando **ls -1 | tee primero.txt | sort -r > segundo.txt**?  ¿Qué diferencia existe con el comando **ls -1 | tee primero.txt | sort -R > segundo.txt**?  Es necesario que explique, paso a paso, qué hace cada uno de los comandos. |

El primer comando lo que hace es copiar el resultado de ls -1 en el archivo primero.txt, y luego ordena el resultado a la inversa y lo almacena en otro archivo llamado segundo.txt.







La diferencia que hay entre el primer comando y el segundo es que en el segundo sale el comando sort -R, en mayúscula, y eso quiere decir que ordena los elementos de manera aleatorio.

El comando ls -1 lista los archivos y directorios en una sola columna en vez de varias.

El comando tee copia la salida del comando anterior en un archivo nuevo o existente llamado primero.txt.

El comando sort -r lo que hace es ordenar los elementos de manera invertida y lo almacena en el archivo segundo.txt.

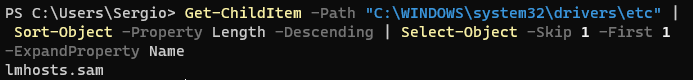
|  |  |
| --- | --- |
| 04 | **Linux**:  Escriba un comando que muestre el nombre del segundo archivo más grande del directorio **/etc**.  **PowerShell**:  Escriba un comando que muestre el nombre del segundo archivo más grande del directorio **C:\WINDOWS\system32\drivers\etc**. |

En Linux:

Comando para mostrar el segundo archivo más grande del directorio /etc.



En PowerShell:



|  |  |
| --- | --- |
| 05 | En **Linux**, escriba un comando que una dos archivos de texto, uno a continuación del otro, en un tercero. |

El comando para unir el contenido de archivos en otro es **cat archivo1.txt archivo2.txt > archivo3.txt**.



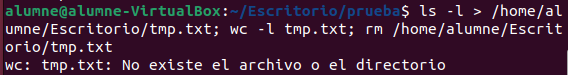
|  |  |
| --- | --- |
| 06 | ¿Cómo podríamos copiar un archivo sin utilizar el comando cp, **utilizando las redirecciones** de la entrada y salida estándar?  Copie el archivo **C:\WINDOWS\system32\drivers\etc\hosts** en su directorio de usuario. |

El comando similar al uso del comando cp sería **cat hosts > *rutaCompletaArchivo***.



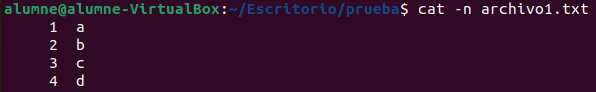
|  |  |
| --- | --- |
| 07 | En **Linux**, ¿Cómo podríamos construir una pipe o tubería (|) con los redirectores de la entrada y salida estándar (< >)? Hay que utilizar un archivo temporal que debemos borrar después de su uso.  Escriba el equivalente a **ls -l | wc -l**. |

La forma sería la siguiente:



|  |  |
| --- | --- |
| 08 | En **Linux**, escriba un comando basado en **cat** que tenga el mismo comportamiento que **nl**. |

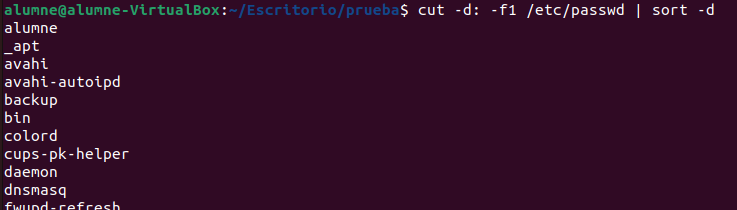
El comando pertinente es **cat -n archivo1.txt.**



|  |  |
| --- | --- |
| 09 | **Linux**:  Escriba un comando que muestre un listado de todos los usuarios del sistema (ÚNICAMENTE el nombre), ordenados por orden alfabético. ***Pista****: Debe trabajar con el archivo /etc/passwd*.  **PowerShell**:  Escriba un comando que muestre un listado de todos los procesos del sistema (ÚNICAMENTE el nombre), ordenados por número de PID. |

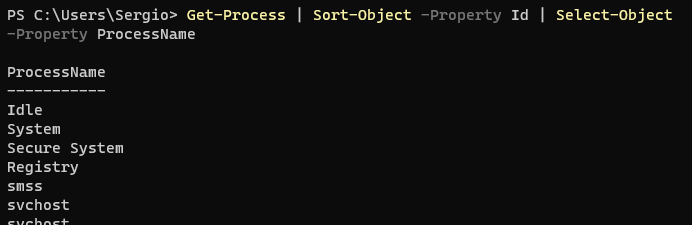
En Linux:

El comando **cut -d: -f1 /etc/passwd** muestra un listado de los usuarios del sistema.



En PowerShell:

El comando para mostrar el listado de todos los procesos del sistema es **Get-Process | Sort-Object -Property Id | Select-Object -Property ProcessName**.



|  |  |
| --- | --- |
| 10 | **Linux**:  Intente crear un directorio en la raíz de su sistema ( / ) y envíe la **salida de error** de este comando a un archivo de su directorio de usuario. ¿Qué ocurre? Pruebe el mismo con la **salida estándar**, ¿qué ocurre? ¿Cómo podríamos hacer que esta **salida de error NO se mostrara por pantalla ni se guardas en ningún archivo**?  **PowerShell**:  Intente listar un archivo que no exista y almacene la **salida de error** a un archivo ‘error.txt’.  ¿Cómo podríamos hacer que esta **salida de error NO se mostrara por pantalla ni se guardas en ningún archivo**? |

En Linux:





No tengo permisos para crear la carpeta ya que se trata de la raíz del sistema y no está permitido.

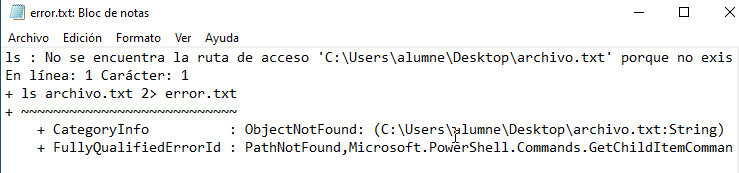
Con la salida estándar me sigue diciendo que me ha denegado la operación, pero tampoco me añade el error al archivo que le he indicado, ya que la redirección usada no lo hace. Crea el archivo, pero vacío.



En PowerShell:

Listo un archivo del escritorio y mando la salida de error al archivo error.txt.





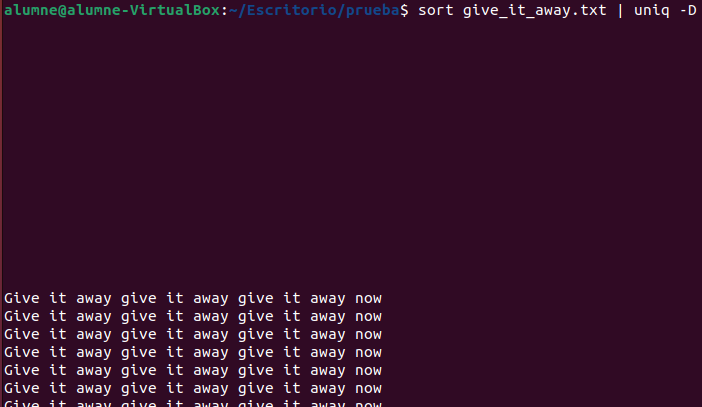
Con **$null** evitamos que se muestre el error en pantalla y que este se guarde en un archivo.

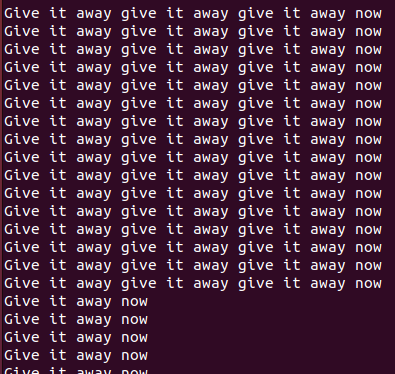


|  |  |
| --- | --- |
| 11 | **Linux**:  Ejecute un comando que le muestre **ÚNICAMENTE las líneas repetidas**, de forma consecutiva, del archivo **give\_it\_away.txt**.  Ejecute un comando que le muestre el número de repeticiones de cada uno de las líneas del archivo **give\_it\_away.txt**.  **PowerShell**:  Ejecute un comando que muestre todos los tipos de comandos (debe fijarse con la columna ‘***Source’***) que encontramos en PowerShell (utilice el comando **Get-Command**). |

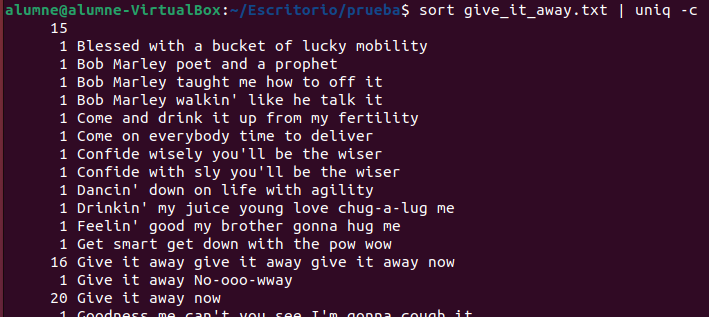
En Linux:

El comando para mostrar las líneas repetidas de forma consecutiva es **sort give\_it\_a\_way.txt | uniq –D**.



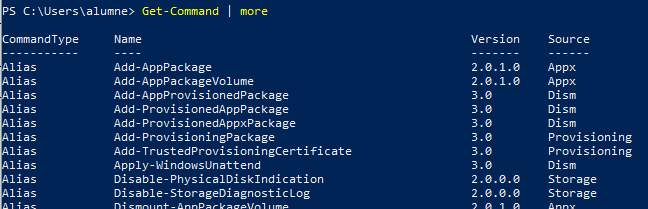


Con la terminación **uniq -c** nos cuenta las veces que están repetidas las líneas.



En PowerShell:

El comando **Get-Command** muestra todos los comandos.



|  |  |
| --- | --- |
| 12 | En **Linux**, utilice el comando paste para unir tres archivos. Haga que todos los tabuladores devueltos se conviertan en espacios. Realice toda esta interacción en una única ejecución. |

Primero muestro el contenido que tiene cada uno de los tres archivos.







Al usar el siguiente comando me junta el contenido de los archivos y me cambia las tabulaciones por un espacio.

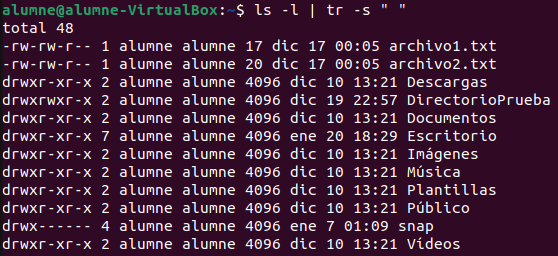


|  |  |
| --- | --- |
| 13 | En **Linux**, ¿Qué opción del comando tr permite comprimir **caracteres repetidos consecutivamente** en uno solo? Pruebe este comando con las “o” de la siguiente cadena "boomboompooooowoooooot". ¿Cómo lo haríamos si en vez de querer comprimir caracteres repetidos lo hiciéramos para **comprimir espacios consecutivos** (comprimir pues los espacios de una salida del comando ‘ls -l’)? |

El comando para comprimir en uno los caracteres repetidos consecutivamente es el siguiente.



Para en vez de hacerlo con caracteres hacerlo con espacios sería de la siguiente manera.



|  |  |
| --- | --- |
| 14 | En **Linux**, escriba un comando que, **para cada uno de los directorios de la raíz del sistema**, cree un archivo de texto con su nombre en el interior (en formato texto). Los archivos de texto generados tendrán que empezar con el prefijo **'carpeta\_'**. |

|  |  |
| --- | --- |
| 15 | En **Linux**, escriba una línea, para cada apartado, que le permitan:  a) Crear un directorio y, si se ha podido crear correctamente, mostrar el mensaje "directorio creado correctamente".  b) Crear un directorio y, si no se ha podido crear correctamente, mostrar el mensaje "no se ha podido crear el directorio".  Para comprobar el funcionamiento de estos apartados intente crear las carpetas en su directorio de usuario y/o en la raíz del sistema. |

Para crear un directorio y lanzar un mensaje de que se ha realizado correctamente se usa lo siguiente:



Para crear un directorio y lanzar un mensaje de que NO se ha realizado correctamente se usa lo siguiente:

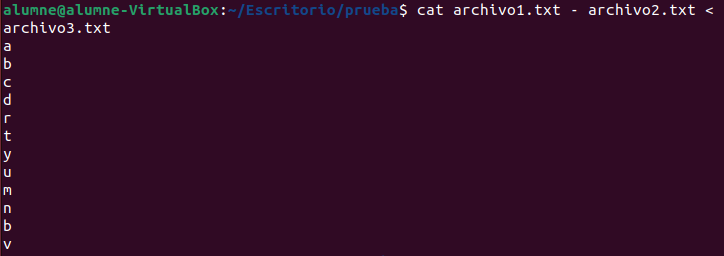


|  |  |
| --- | --- |
| 16 | En **Linux**, ¿Qué efecto tiene el comando **echo “------” | cat archivo1 - archivo2**? ¿Qué efecto tiene el comando **cat archivo1 - archivo2 < archivo3**? |

El comando **echo “------” | cat archivo1 - archivo2** primero muestra el contenido de archivo1 seguido de una línea y finalmente el contenido del archivo2.



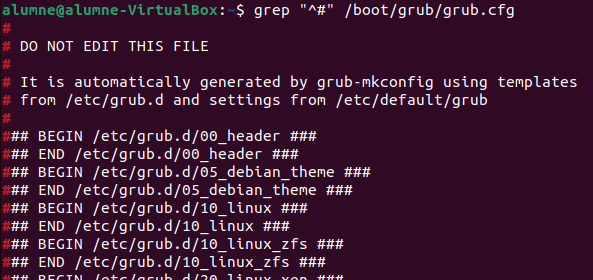
El comando **cat archivo1 – archivo2 < archivo3** concatena el contenido de los tres archivos, primero muestra el contenido del archivo1, luego el del archivo3 y finalmente el del archivo2.



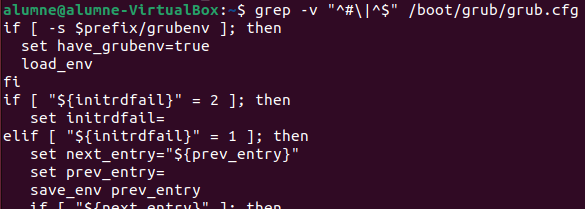
|  |  |
| --- | --- |
| 17 | **Linux**:  Los archivos de configuración suelen indicar los comentarios con líneas que comienzan con **#**. Escriba un comando que muestre todas las líneas de comentarios del archivo **grub.cfg** (indique la ruta completa).  Escriba un comando que muestre el contenido del archivo **grub.cfg** eliminando los comentarios y también las líneas vacías.  **Pista**: *una línea vacía es aquella que no contiene ningún carácter*.  **PowerShell**:  Escriba un comando que muestre el contenido del archivo ***C:\WINDOWS\system32\drivers\etc\networks*** eliminando los comentarios y también las líneas vacías. |

En Linux:

El siguiente comando muestra los comentarios del archivo grub.cfg.



El siguiente es el comando para mostrar el contenido del grub.cfg sin los comentarios y sin líneas vacías.



En PowerShell:

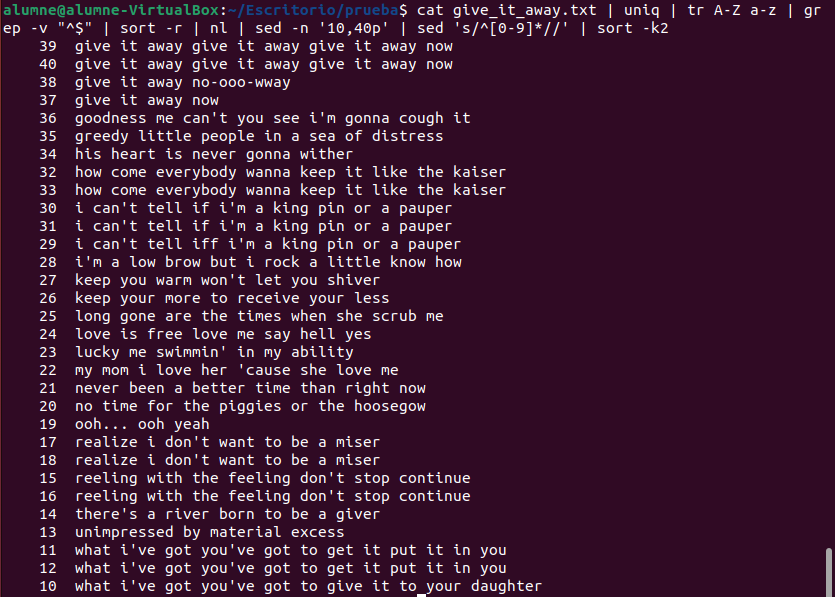
El comando correcto es el siguiente.



|  |  |
| --- | --- |
| 18 | Tome el archivo de texto **give\_it\_away.txt** y realice las siguientes acciones (en el mismo orden, comunicadas con tuberías o pipas):  **Linux**:  1. Elimine las líneas repetidas consecutivas.  2. Cambie todas las letras mayúsculas que haya por minúsculas.  3. Elimine todas las líneas vacías existentes.  4. Ordene las líneas resultantes en orden alfabético inverso.  5. Numere todas las líneas.  6. Quédese sólo con las líneas que van de la 10 a la 40.  7. Elimine la numeración de las líneas que había añadido al paso 5.  8. Reordenar las líneas en orden alfabético por la segunda palabra de cada línea.  **PowerShell**:  1. Elimine las líneas repetidas consecutivas.  2. Elimine todas las líneas vacías existentes.  3. Numere todas las líneas.  4. Quédese sólo con las líneas que van de la 10 a la 40.  5. Elimine la numeración de las líneas que había añadido al paso 5.  6. Ordene las líneas resultantes en orden alfabético inverso.  **Explique cada uno de los pasos que haya utilizado en este comando.** |

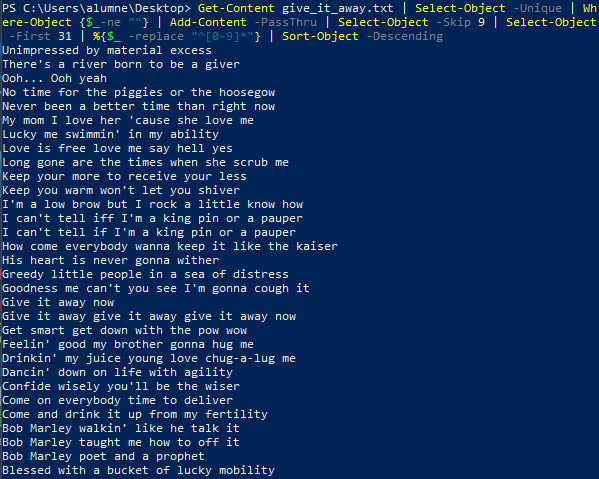
En Linux:

1. **Cat give\_it\_away.txt**: muestra el contenido del archivo.
2. **Uniq**: elimina las líneas repetidas consecutivamente.
3. **Tr A-Z a-z**: convierte las letras mayúsculas en minúsculas.
4. **Grep -v “^$”**: elimina las líneas vacías del documento.
5. **Sort -r**: invierte alfabéticamente el orden de las líneas.
6. **Nl**: numera las líneas del documento.
7. **Sed -n ’10,40p’**: muestra únicamente las líneas de la 10 a la 40.
8. **Sed ‘s/^[0-9]\*//’**: elimina la numeración de las líneas.
9. **Sort -k2**: reordena las líneas en orden alfabético por la segunda palabra de cada línea.



En PowerShell:

1. **Get-Content give\_it\_away.txt**: muestra el contenido del archivo.
2. **Select-Object -Unique**: elimina las líneas consecutivas repetidas
3. **Where-Object {$\_ -ne “”}**: elimina las líneas vacías existentes.
4. **Add-Content -PassThru**: añade numeración a cada línea.
5. **Select-Object -Skip 9 y Select-Object -Skip 31**: se queda sólo con las líneas que van de la 10 a la 40.
6. **Nl**: numera las líneas del documento.
7. **%{$\_ -replace “^[0-9]\*”}**: elimina la numeración de las líneas.
8. **Sort-Object -Descending**: ordena las líneas resultantes en orden alfabético inverso.

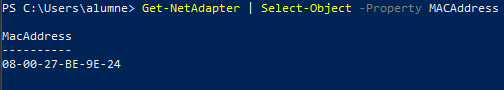


|  |  |
| --- | --- |
| 19 | Tanto en **Linux** como en **Windows**, escriba un comando que devuelva la MAC de la nuestra tarjeta de red. Para ello, será necesario que:  1. Ejecute el comando **ifconfig** (**'ipconfig /all'** o **'Get-NetAdapter'** en PowerShell).  2. Seleccione de alguna manera la línea que contiene la MAC de su tarjeta de red.  **Nota**: En **Linux** seleccione únicamente el valor de la MAC, en **Windows** podemos obtener toda la línea entera.  **Explique cada uno de los pasos que haya utilizado en este comando.** |

En Linux:

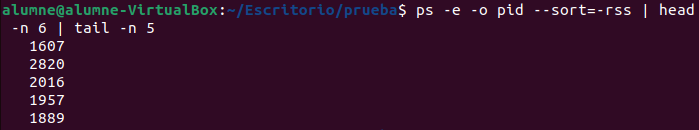


En PowerShell:

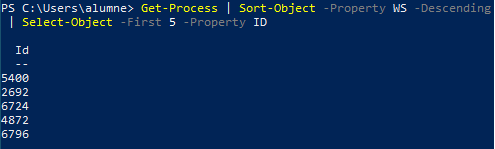


|  |  |
| --- | --- |
| 20 | Tanto en **Linux** como en **Windows**, escriba un comando que muestre ÚNICAMENTE los PID de los 5 procesos que consumen mayor memoria del sistema.  **Nota**: en el caso de Windows nos referimos a la **memoria paginada**. |

En Linux:



En Windows:



|  |  |
| --- | --- |
| 21 | Escriba un comando que permita mostrar por pantalla el siguiente mensaje: **\*\*\*MICRO$OFT RULE$\*\*\***. |

Con el comando **echo** podemos imprimir el mensaje que queremos por pantalla.



|  |  |
| --- | --- |
| 22 | En **Linux**, ¿Para qué sirve cada una de las expresiones regulares de **grep** que se indican a continuación? |

|  |  |
| --- | --- |
| **COMANDO** | **RESULTADO** |
| ^ | Se usa para buscar líneas que empiecen con uno o más caracteres indicados después del símbolo. |
| $ | Utilizado para buscar líneas que terminen con uno o más caracteres indicados antes del símbolo. |
| [...] | Se usa para buscar líneas que contengan al menos uno de los caracteres que haya dentro de los corchetes. |
| [^...] | Utilizado para buscar líneas que no contengan ninguno de los caracteres que haya dentro de los corchetes. |
| [m-r] | Se utiliza para buscar líneas que contengan al menos uno de los caracteres que se encuentren dentro del rango de m y r. |
| . | Sirve para buscar una línea que contenga cualquier carácter. El “.” es un comodín que representa cualquier carácter. |
| c\* | Se usa para buscar una línea que contenga cero o más instancias del carácter "c". |
| .\* | Busca cualquier línea que contenga cualquier carácter, ya sea una o varias veces. |
| \{n\} | Sirve para buscar una línea que contenga una cantidad específica de un carácter o conjunto de caracteres consecutivamente. |
| \{n,\} | Se usa para especificar una cantidad mínima de veces que un carácter o conjunto de caracteres debe aparecer. |
| \{n,m\} | Se utiliza para especificar un rango mínimo de apariciones y un número máximo de apariciones (por ejemplo: grep “a{3,5}”, donde 3 es el mínimo y 5 es el máximo.) |
| \ | Utilizado en las expresiones regulares para escapar caracteres especiales (por ejemplo: grep “a\\*b, busca una cadena que tenga una a, seguido de un \* y de una b). |

|  |  |
| --- | --- |
| 23 | En **Linux**, explique qué hace cada uno de los siguientes comandos: |

|  |  |
| --- | --- |
| **COMANDO** | **RESULTADO** |
| grep give \* | Muestra las líneas que tienen la contengan give. |
| grep -l give \* | Muestra el nombre de los archivos los cuales en su contenido hay alguna cadena “give”. |
| grep -i give \*.txt | Busca en todas las líneas de todos los archivos .txt la palabra “give”, indistintamente de si están en mayúscula o minúscula, y las imprime. |
| grep -i -h give \*.txt | Hace lo mismo que el anterior, pero elimina el nombre del archivo del principio de la línea y el delimitador. |
| grep -n give give\_it\_away.txt | Busca una cadena de texto específica en un archivo y muestra las líneas en las que se encuentra esa cadena, junto con el número de línea. |
| grep -cv give give\_it\_away.txt | Busca una cadena de texto en un archivo y muestra la cantidad de veces que no se encuentra la cadena en el archivo. |
| grep give$ give\_it\_away.txt | Muestra las líneas en las que las que “give” sea el ultimo carácter de la línea. |
| grep ^give$ \* | Muestra solo las líneas que contengan solo la palabra “give” y no otras palabras o caracteres adicionales. |
| grep [A-Z] give\_it\_away.txt | Busca cualquier carácter en mayúsculas en el archivo "give\_it\_away.txt". |
| grep 'Stal\{1,2\}man' \* |  |
| grep '^error' \*.log > problemes.txt  grep '^dump' \*.log >> problemes.txt | El comando "grep '^error' \*.log > problemes.txt" busca la cadena "error" al principio de cualquier línea en todos los archivos con extensión .log del directorio actual y redirige el resultado a un archivo llamado "problemes.txt" en el directorio actual.  El comando "grep '^dump' \*.log >> problemes.txt" busca la cadena "dump" al principio de cualquier línea en todos los archivos con extensión .log del directorio actual y agrega el resultado a "problemes.txt" existente en el directorio actual. |
| grep -c /bin/bash /etc/passwd | Busca la cadena "/bin/bash" en el archivo "/etc/passwd" y cuenta el número de veces que se encuentra esa cadena. |