Sistemas Operativos
Tarea #1
Shell Linux

Escuela de Ingeniería Civil Informática
Universidad de Valparaíso

Katerina Peñaloza Caballería katerina.penaloza@alumnos.uv.cl

01-10-2021

1) Estudie y explique para qué sirven los comandos **ls**, **cat**, **chmod**, **echo**, **grep**, **cp**, **mv**, **rm** y **wc**. Dé ejemplos de uso. Para el comando **ls** averigüe para qué sirven las opciones -l, -t y -a. De ejemplo de uso para cada uno de esas opciones y la combinación de ellas. Ayuda: use el comando man, ej. **man ls**.

   a) **ls** = lista los ficheros contenidos en un directorio, si no se especifica el directorio, por defecto listará el directorio actual.
   -l → utiliza una lista con formato largo, en donde se muestran permisos del fichero, el número de enlace, nombre del propietario, nombre del grupo al que pertenece, tamaño en bytes, una marca de tiempo (fecha de última modificación) y nombre del fichero.
   -t → muestra los ficheros por orden de fecha de última modificación, los más nuevos primero.
   -a → lista todos los ficheros de un directorio, incluidos ficheros ocultos que empiezan con punto (.)

   Ejemplos: Ejemplos de los archivos del directorio /proc ya que es donde hay más cantidad de archivos.

   ls sin opciones ni directorio especificado → muestra todos los ficheros de ./proc

```
katzewy@katzewy:/proc$ ls
1     160   2504  533  653  86        bus          ioports      mounts        sysvipc
10    1689  277   534  658  87        cgroups      irq          mtrr          thread-self
105   17    278   544  688  88        cmdline      kallsyms     net           timer_list
108   18    2985  546  70   89        consoles     kcore        pagetypeinfo  tty
11    19    2986  548  71   9         cpuinfo      keys         partitions    uptime
1116  191   3     554  717  90        crypto       key-users    pressure      version
1154  192   3091  555  72   91        devices      kmsg         sched_debug   version_signature
12    2     3112  570  73   93        diskstats    kpagecgroup  schedstat     vmallocinfo
121   20    3145  6    74   95        dma          kpagecount   scsi          vmstat
13    207   352   613  75   96        driver       kpageflags   self          zoneinfo
14    21    382   615  76   968       execdomains  loadavg      slabinfo
15    22    4     632  77   969       fb           locks        softirqs
156   23    402   633  78   977       filesystems  mdstat       stat
1566  238   530   640  79   acpi      fs           meminfo      swaps
157   24    531   641  83   asound    interrupts   misc         sys
16    2448  532   645  84   buddyinfo iomem        modules      sysrq-trigger
```

   ls ./sys → muestra todos los ficheros contenidos en ./sys

```
katzewy@katzewy:/proc$ ls ./sys
abi  debug  dev  fs  kernel  net  user  vm
```

ls -l → muestra la lista con formato largo de los ficheros contenidos en ./proc

```
-r--------  1 root             root                             0 oct  1 22:13 kpagecgroup
-r--------  1 root             root                             0 oct  1 22:13 kpagecount
-r--------  1 root             root                             0 oct  1 22:13 kpageflags
-r--r--r--  1 root             root                             0 oct  1 21:45 loadavg
-r--r--r--  1 root             root                             0 oct  1 22:13 locks
-r--r--r--  1 root             root                             0 oct  1 21:45 mdstat
-r--r--r--  1 root             root                             0 oct  1 21:45 meminfo
-r--r--r--  1 root             root                             0 oct  1 22:13 misc
-r--r--r--  1 root             root                             0 oct  1 22:13 modules
lrwxrwxrwx  1 root             root                            11 oct  1 21:45 mounts -> self/mounts
-rw-r--r--  1 root             root                             0 oct  1 21:45 mtrr
lrwxrwxrwx  1 root             root                             8 oct  1 21:45 net -> self/net
-r--------  1 root             root                             0 oct  1 22:13 pagetypeinfo
-r--r--r--  1 root             root                             0 oct  1 21:45 partitions
dr-xr-xr-x  5 root             root                             0 oct  1 22:13 pressure
-r--r--r--  1 root             root                             0 oct  1 22:13 sched_debug
-r--r--r--  1 root             root                             0 oct  1 22:13 schedstat
dr-xr-xr-x  5 root             root                             0 oct  1 21:45 scsi
lrwxrwxrwx  1 root             root                             0 oct  1 21:45 self -> 3285
-r--------  1 root             root                             0 oct  1 22:13 slabinfo
-r--r--r--  1 root             root                             0 oct  1 22:13 softirqs
-r--r--r--  1 root             root                             0 oct  1 21:46 stat
-r--r--r--  1 root             root                             0 oct  1 21:45 swaps
dr-xr-xr-x  1 root             root                             0 oct  1 21:45 sys
--w-------  1 root             root                             0 oct  1 21:45 sysrq-trigger
dr-xr-xr-x  5 root             root                             0 oct  1 22:13 sysvipc
lrwxrwxrwx  1 root             root                             0 oct  1 21:45 thread-self -> 3285/tas
k/3285
-r--------  1 root             root                             0 oct  1 22:13 timer_list
dr-xr-xr-x  6 root             root                             0 oct  1 21:52 tty
-r--r--r--  1 root             root                             0 oct  1 21:45 uptime
-r--r--r--  1 root             root                             0 oct  1 21:45 version
-r--r--r--  1 root             root                             0 oct  1 22:13 version_signature
-r--------  1 root             root                             0 oct  1 22:13 vmallocinfo
-r--r--r--  1 root             root                             0 oct  1 22:13 vmstat
-r--r--r--  1 root             root                             0 oct  1 22:13 zoneinfo
katzewy@katzewy:/proc$ _
```

ls -t → muestra los ficheros contenidos en ./proc por orden de última fecha de modificación

```
katzewy@katzewy:/proc$ ls -t
3286          kpagecgroup        tty          548          thread-self  10   2    75       swaps
3284          kpagecount         1116         554          asound       105  20   76       partitions
3277          kpageflags         1566         555          bus          108  207  77       uptime
3274          locks              1689         loadavg      fs           11   21   78       devices
3235          misc               2448         cpuinfo      irq          12   22   88       mounts
3179          modules            977          717          kallsyms     121  23   84       1
buddyinfo     pagetypeinfo       2504         688          kmsg         13   233  86       filesystems
consoles      pressure           stat         658          mtrr         14   24   87       kcore
crypto        sched_debug        1154         interrupts   scsi         15   277  88       sys
diskstats     schedstat          969          653          sysrq-trigger 156 278  89       cmdline
dma           slabinfo           968          645          acpi         157  3    9
driver        softirqs           402          641          570          16   4    90
execdomains   sysvipc            530          640          534          160  6    91
fb            timer_list         531          683          mdstat       17   70   93
iomem         version_signature  532          682          version      18   71   95
ioports       vmallocinfo        533          615          382          19   72   96
keys          vmstat             544          613          meminfo      191  73   cgroups
key-users     zoneinfo           546          self         352          192  74   net
```

ls -a → se puede apreciar como se muestran los ficheros **“.”** y **“..”**

```
katzewy@katzewy:/proc$ ls -a
.          157   24    530  640  83    asound        interrupts   misc          sys
..         16    2448  531  641  84    buddyinfo     iomem        modules       sysrq-trigger
1          160   2504  532  645  86    bus           ioports      mounts        sysvipc
10         1689  277   533  658  87    cgroups       irq          mtrr          thread-self
105        17    278   534  658  88    cmdline       kallsyms     net           timer_list
108        18    3     544  688  89    consoles      kcore        pagetypeinfo  tty
11         19    3235  546  70   9     cpuinfo       keys         partitions    uptime
1116       191   3274  548  71   90    crypto        key-users    pressure      version
1154       192   3277  554  717  91    devices       kmsg         sched_debug   version_signature
12         2     3284  555  72   93    diskstats     kpagecgroup  schedstat     vmallocinfo
121        20    3288  570  73   95    dma           kpagecount   scsi          vmstat
13         207   3289  6    74   96    driver        kpageflags   self          zoneinfo
14         21    352   613  75   968   execdomains   loadavg      slabinfo
15         22    382   615  76   969   fb            locks        softirqs
156        23    4     632  77   977   filesystems   mdstat       stat
1566       233   402   683  78   acpi  fs            meminfo      swaps
katzewy@katzewy:/proc$
```

ls -lt → muestra la lista con formato largo de los ficheros contenidos en ./proc por orden de fecha de última modificación.

```
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 70
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 71
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 72
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 73
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 74
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 75
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 76
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 77
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 78
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 83
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 84
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 86
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 87
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 88
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 89
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 9
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 90
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 91
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 93
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 95
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 96
-r--r--r--  1 root             root                           0 oct  1 21:45 cgroups
lrwxrwxrwx  1 root             root                           8 oct  1 21:45 net -> self/net
-r--r--r--  1 root             root                           0 oct  1 21:45 swaps
-r--r--r--  1 root             root                           0 oct  1 21:45 partitions
-r--r--r--  1 root             root                           0 oct  1 21:45 uptime
-r--r--r--  1 root             root                           0 oct  1 21:45 devices
lrwxrwxrwx  1 root             root                          11 oct  1 21:45 mounts -> self/mounts
dr-xr-xr-x  9 root             root                           0 oct  1 21:45 1
-r--r--r--  1 root             root                           0 oct  1 21:45 filesystems
-r--------  1 root             root           140737477885952 oct  1 21:45 kcore
dr-xr-xr-x  1 root             root                           0 oct  1 21:45 sys
-r--r--r--  1 root             root                           0 oct  1 21:45 cmdline
katzewy@katzewy:/proc$
```

ls -la → se muestran los archivos con formato, incluyendo los ocultos en /proc/sys

```
katzewy@katzewy:/proc$ cd ./sys
katzewy@katzewy:/proc/sys$ ls -a
.  ..  abi  debug  dev  fs  kernel  net  user  vm
katzewy@katzewy:/proc/sys$ ls -la
total 0
dr-xr-xr-x   1 root root 0 oct  1 21:45 .
dr-xr-xr-x 158 root root 0 oct  1 21:45 ..
dr-xr-xr-x   1 root root 0 oct  1 22:27 abi
dr-xr-xr-x   1 root root 0 oct  1 22:27 debug
dr-xr-xr-x   1 root root 0 oct  1 22:27 dev
dr-xr-xr-x   1 root root 0 oct  1 21:45 fs
dr-xr-xr-x   1 root root 0 oct  1 21:45 kernel
dr-xr-xr-x   1 root root 0 oct  1 21:45 net
dr-xr-xr-x   1 root root 0 oct  1 22:27 user
dr-xr-xr-x   1 root root 0 oct  1 21:45 vm
katzewy@katzewy:/proc/sys$ _
```

ls -ta → muestra todos los ficheros (incluidos los ocultos) contenidos en /proc/sys por orden de fecha

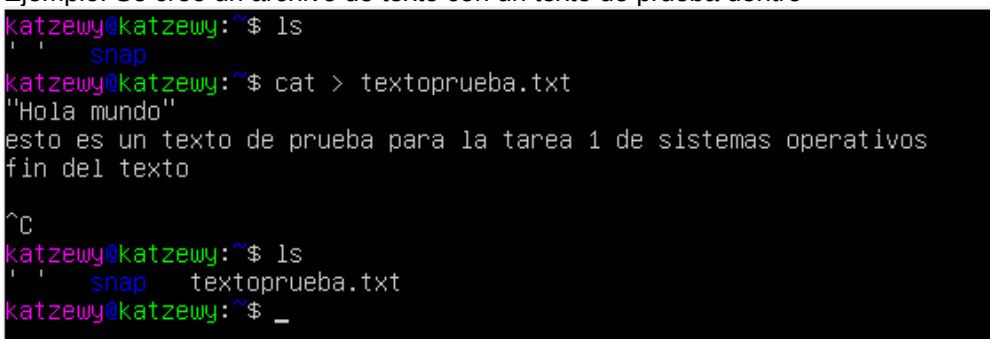
```
katzewy@katzewy:/proc/sys$ ls -at
abi  debug  dev  user  ..  vm  net  fs  .  kernel
katzewy@katzewy:/proc/sys$
```

ls -lat → muestra la lista con formato largo de los ficheros, incluidos los ocultos, contenidos en /proc/sys/ por orden de fecha de última modificación.

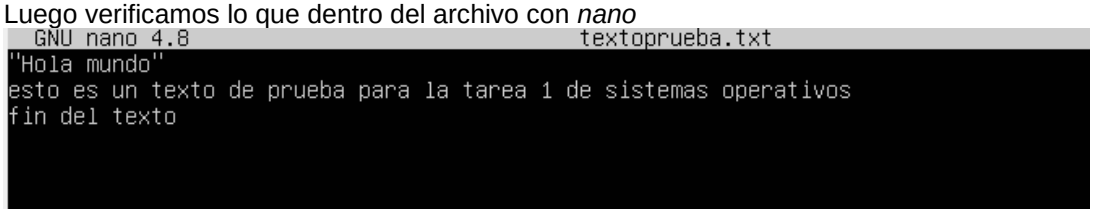
```
katzewy@katzewy:/proc/sys$ ls -lat
total 0
dr-xr-xr-x   1 root root 0 oct  1 22:27 abi
dr-xr-xr-x   1 root root 0 oct  1 22:27 debug
dr-xr-xr-x   1 root root 0 oct  1 22:27 dev
dr-xr-xr-x   1 root root 0 oct  1 22:27 user
dr-xr-xr-x 159 root root 0 oct  1 21:45 ..
dr-xr-xr-x   1 root root 0 oct  1 21:45 vm
dr-xr-xr-x   1 root root 0 oct  1 21:45 net
dr-xr-xr-x   1 root root 0 oct  1 21:45 fs
dr-xr-xr-x   1 root root 0 oct  1 21:45 .
dr-xr-xr-x   1 root root 0 oct  1 21:45 kernel
katzewy@katzewy:/proc/sys$ _
```

b) **cat** = permite crear, fusionar o imprimir archivos en la pantalla de salida estándar o en otro archivo

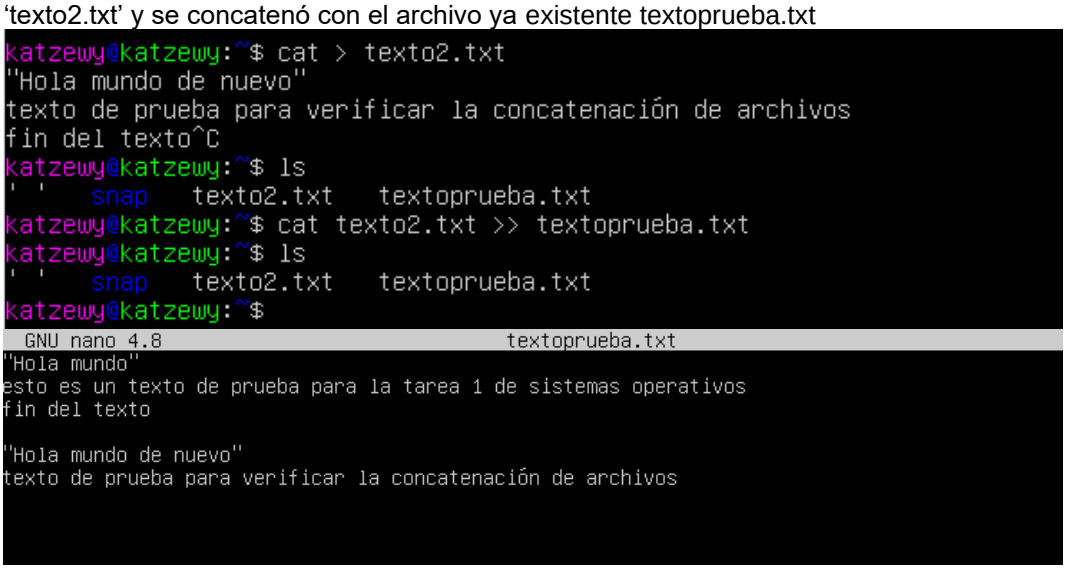
Ejemplo: Se creó un archivo de texto con un texto de prueba dentro



Luego verificamos lo que dentro del archivo con *nano*



También para verificar la concatenación de archivos se creó un archivo nuevo llamado 'texto2.txt' y se concatenó con el archivo ya existente textoprueba.txt



c) chmod = permite cambiar los permisos de lectura, escritura y ejecución de un fichero o directorio y se usa para definir la forma en que se puede acceder a un archivo**.**
Su sintaxis es: chmod [opciones] permisos "nombre del archivo"
Los permisos básicos son: lectura - **r**, escritura - **w** y ejecución – **x**
Los usuarios a los que se les puede asignar permisos son: usuario – **u,** grupo – **g**, otros – **o**, todos – **a.**
Hay dos formas de representar estos permisos: con símbolos (caracteres alfanuméricos) o con números octales (dígitos del 0 al 7).

Ejemplo con símbolos

Supongamos que el usuario puede leer, escribir y ejecutar el archivo, el grupo puede leer y ejecutar el archivo y los otros solo pueden leer el archivo:

```
katzewy@katzewy:~$ ls
' '   snap   texto2.txt   textoprueba.txt
katzewy@katzewy:~$ chmod u=rwx,g=rx,o=r textoprueba.txt
katzewy@katzewy:~$ ls -l
total 16
-rw-rw-r-- 1 katzewy katzewy   12 sep  3 15:21 ' '
drwxr-xr-x 3 katzewy katzewy 4096 oct  1 21:50  snap
-rw-rw-r-- 1 katzewy katzewy   83 oct  1 23:26  texto2.txt
-rwxr-xr-- 1 katzewy katzewy  177 oct  1 23:28  textoprueba.txt
```

Luego les quitamos todos los permisos a grupo y otros:

```
katzewy@katzewy:~$ chmod go= textoprueba.txt
katzewy@katzewy:~$ ls -l
total 16
-rw-rw-r-- 1 katzewy katzewy   12 sep  3 15:21 ' '
drwxr-xr-x 3 katzewy katzewy 4096 oct  1 21:50  snap
-rw-rw-r-- 1 katzewy katzewy   83 oct  1 23:26  texto2.txt
-rwx------ 1 katzewy katzewy  177 oct  1 23:28  textoprueba.txt
katzewy@katzewy:~$
```

Ejemplo con números octales

El mismo ejemplo anterior, pero con números octales:

```
katzewy@katzewy:~$ ls -l
total 16
-rw-rw-r-- 1 katzewy katzewy   12 sep  3 15:21 ' '
drwxr-xr-x 3 katzewy katzewy 4096 oct  1 21:50  snap
-rw-rw-r-- 1 katzewy katzewy   83 oct  1 23:26  texto2.txt
-rwx------ 1 katzewy katzewy  177 oct  1 23:28  textoprueba.txt
katzewy@katzewy:~$ chmod 754 texto2.txt
katzewy@katzewy:~$ ls -l
total 16
-rw-rw-r-- 1 katzewy katzewy   12 sep  3 15:21 ' '
drwxr-xr-x 3 katzewy katzewy 4096 oct  1 21:50  snap
-rwxr-xr-- 1 katzewy katzewy   83 oct  1 23:26  texto2.txt
-rwx------ 1 katzewy katzewy  177 oct  1 23:28  textoprueba.txt
katzewy@katzewy:~$
```

c) **echo** = muestra una línea de texto en el output, se pueden imprimir números, variables, strings, operaciones y mucho más.

Ejemplo con strings y variables:

```
katzewy@katzewy:~$ nombre='Katerina Peñaloza'
katzewy@katzewy:~$ edad=21
katzewy@katzewy:~$ echo Hola, soy $nombre y tengo $edad años
Hola, soy Katerina Peñaloza y tengo 21 años
```

d) **grep** = toma una expresión regular de la línea de comandos e imprime las líneas que coincidan con ciertos patrones indicados.
Su sintaxis es: grep [opciones] [expresión regular] [nombrearchivo]
Combinado con otros comandos es útil para filtrar líneas que uno necesita ver

Ejemplo:
Utilizando el archivo creado con el comando cat, mostrar las líneas que contengan la palabra “hola”, utilizando, además -i para ignorar las mayúsculas.

```
katzewy@katzewy:~$ grep -i hola textoprueba.txt
"Hola mundo"
"Hola mundo de nuevo"
katzewy@katzewy:~$ _
```

e) **cp** = permite copiar archivos y directorios.

Ejemplo:
Copiar el archivo textoprueba.txt

```
katzewy@katzewy:~$ ls
'  '   katzewy   snap   texto2.txt   textoprueba.txt
katzewy@katzewy:~$ cp textoprueba.txt textopruebacopia.txt
katzewy@katzewy:~$ ls
'  '   katzewy   snap   texto2.txt   textopruebacopia.txt   textoprueba.txt
```

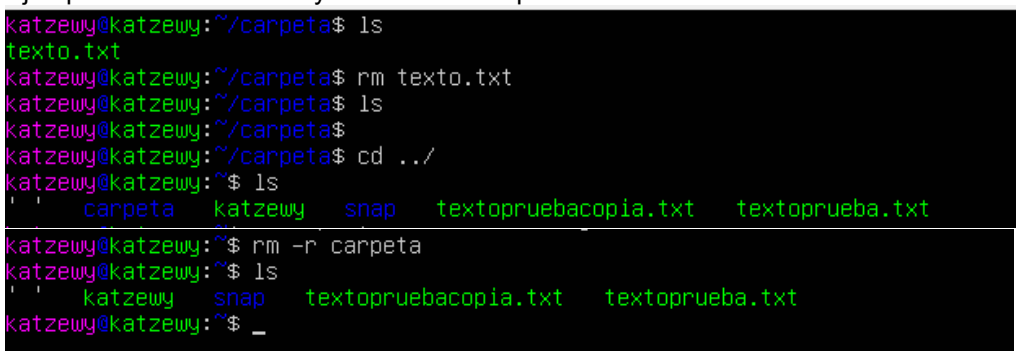
f) **mv** = sirve para mover (renombrar) archivos o directorio, además tiene una opción para crear un respaldo con -backup.

Ejemplo: Renombramos “texto2.txt” a “texto.txt” y luego lo movemos al directorio carpeta

```
katzewy@katzewy:~$ ls
'  '   katzewy   snap   texto2.txt   textopruebacopia.txt   textoprueba.txt
katzewy@katzewy:~$ #renombramos texto2.txt a texto.txt
katzewy@katzewy:~$ mv texto2.txt texto.txt
katzewy@katzewy:~$ ls
'  '   katzewy   snap   textopruebacopia.txt   textoprueba.txt   texto.txt
katzewy@katzewy:~$
katzewy@katzewy:~$ mv texto.txt ./carpeta
katzewy@katzewy:~$ ls
'  '   carpeta   katzewy   snap   textopruebacopia.txt   textoprueba.txt
katzewy@katzewy:~$ cd ./carpeta
katzewy@katzewy:~/carpeta$ ls
texto.txt
```

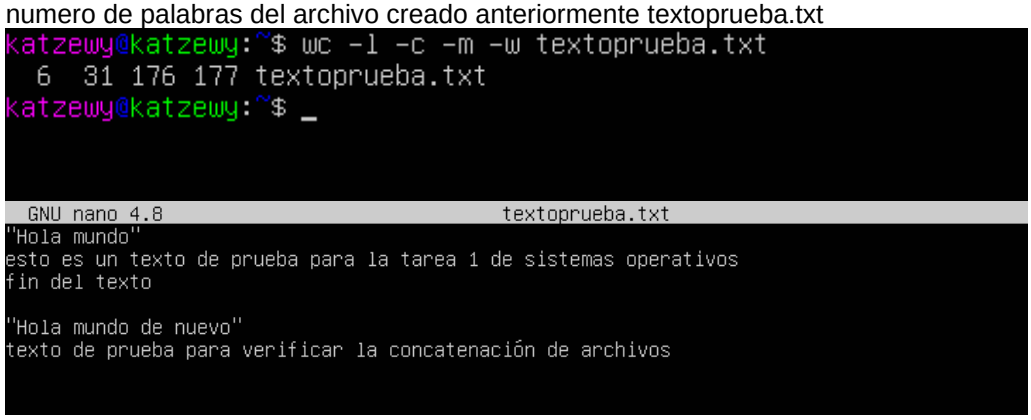
g) **rm** = es un comando para eliminar archivos y directorios.

Ejemplo: borrar “texto.txt” y el directorio carpeta



h) **wc** = es un comando que permite realizar distintos conteos, ya sea de líneas **-l**, caracteres **-m**, bytes **-c** o palabras **-w**.

Ejemplo: se mostrará el numero de líneas, numero de bytes, número de caracteres y numero de palabras del archivo creado anteriormente textoprueba.txt



2) Explique qué son los metacaracteres y dé ejemplos de uso de ellos.

Para empezar una expresión regular son cadenas de caracteres basadas en reglas sintácticas que permiten describir **secuencias de caracteres** que conforman un patrón de búsqueda. Esta expresión regular puede estar conformado por caracteres normales y metacaracteres.

Un metacaracter es un carácter no alfabético que tiene significado especial en las expresiones regulares, describen ciertas construcciones o disposiciones de caracteres, por ejemplo, si un carácter debe estar en el inicio de una línea o palabra, o si solo puede aparecer exactamente una vez, más o menos veces, etc.

El conjunto de metacaracteres que se pueden utilizar en sintaxis de expresión regular ampliada es el siguiente: * + ? $ ^ . () | \ {} [

Se usa por ejemplo en las consultas en bases de datos o también, por ejemplo, cuando uno quiere establecer ciertos patrones que debe tener una contraseña para que sea segura y sobre todo en la Shell de Linux con el comando grep.

Algunos ejemplos básicos de metacaracteres son:

***** = sustituye cualquier secuencia de caracteres, incluido ningún carácter

? = sustituye por cualquier carácter

[ ] = se utiliza para especificar una lista de caracteres o un rango.

^ = indica que la expresión comienza al principio de la línea.

$ = indica el final de línea.

a* = indica cualquier conjunto de caracteres que empiece por a.

*a = indica cualquier conjunto de caracteres que acabe en a.

<, > = redirigir

$ en Shell = obtener el valor de una variable

| = pipeline, representa la conexión entre la salida estándar del comando actual y la entrada estándar del comando siguiente.

3) Explique en que consiste la expansión por paréntesis de conjunto
(Brace Expansion). Con esta herramienta, resuelve el siguiente problema:

> En un directorio, se quieren crear subdirectorios para que almacenen respaldos diarios de todo un año. Debe tener la siguiente estructura:
>
> directorio_actual/
> + 2021-01-01/
> + 2021-01-02/
> + 2021-01-03/
> . . .
> + 2021-09-18/
> . . .
> + 2021-10-18/
> . . .
> + 2021-12-31/
>
> Suponga que todos los meses tienen 31 días. Debe ejecutar UN sólo comando para crear la estructura de directorios solicitada.

Brace Expansion = La expansión de llaves es un mecanismo mediante el cual se pueden generar cadenas arbitrarias. Las cadenas especificadas se utilizan para generar todas las combinaciones posibles con los prefijos y sufijos opcionales. Por lo general, se usa para generar listas de cadenas que se pueden usar en scripts y alias y en la línea de comandos de Linux.

Respecto al problema propuesto el comando a utilizar es:

mkdir 2021-{01..12}-{01-31}

Se especifica más sobre cómo se llegó a esa conclusión en el archivo readme.md

4) La interconexión de comandos a través de *pipes* permite construir, de forma muy simple, nuevas herramientas. Como ejemplo considere los comandos ls y wc, que interconectados permiten contar archivos del directorio actual: ls | wc -l. Mediante el uso de pipes resuelva:

1. Mediante grep, encontrar archivos cuyo nombre contenga el carácter i en el directorio /bin.
2.
3. Contar los archivos con una secuencia de permisos r-x en los directorios /bin y /usr/bin.

5) Las variables de ambiente definen aspectos del entorno de programación, y los comandos set y echo (mediante el metacaracter $) permiten ver su contenido.

1. Investigue el uso de las variables HOME, SHELL, PATH, y PWD. ¿Cómo se puede visualizar su contenido?
2. Investigue cómo se puede modificar el valor de una variable de ambiente (ayuda: investigue el operador '=' y export).
3. Ejecutando el comando *bash* dentro de la línea de comandos se crea un sub-shell, ¿Cómo afecta esto las variables de ambiente? ¿Cuál es el efecto de export? Explique y dé ejemplos

6) El intérprete de comandos bash es también un lenguaje de programación, con estamentos de control de flujo como for, while, etc. El código fuente a menudo se llama *script*. Si el archivo que contiene el script se llama **ejemplo.sh**, el comando **chmod +x ejemplo.sh**, le da permisos de ejecución al archivo **ejemplo.sh**.

Implementar un script BASH que liste cada argumento de entrada por separado, incluyendo el nombre del script. Además, debe mostrar su PID y mostrar las 10 primeras líneas del archivo /proc/PID/status.