

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS Y SISTEMAS
MANEJO E IMPLEMENTACIÓN DE ARCHIVOS A+ y A-
Ing. Álvaro Díaz Ing. Oscar Paz Campos
Aux. Edward Gómez Aux. Georgina Estrada
Segundo Semestre de 2016



FILE SYSTEM EXT2/EXT3

Introducción

El curso de Manejo e Implementación de Archivos busca que el estudiante aprenda los conceptos sobre la administración de archivos, tanto en hardware como software, sistemas de archivos, particiones, entre otros conceptos, para después introducirse en las bases de datos. El proyecto busca que el estudiante aplique estos conceptos y pueda aprenderlos implementando su funcionalidad.

Objetivos

- Aprender a administrar archivos y escribir estructuras en C
- Comprender el sistema de archivos EXT2/EXT3
- Aplicar el formato rápido y completo en una partición
- Crear una aplicación de comandos
- Aplicar la teoría de ajustes
- Aplicar la teoría de particiones
- Utilizar GraphViz para mostrar reportes
- Restringir y administrar el acceso a los archivos y carpetas en ext2/ext3 por medio de usuarios.
- Administrar los usuarios y permisos por medio de grupos

ÍNDICE

Introducción	1
Objetivos.....	1
Comandos.....	4
Administración de Discos.....	5
mkdisk	5
rmdisk	6
fdisk.....	7
mount.....	10
umount.....	10
disk free (df)	11
disk used (du)	12
mkfs.....	13
Discos	14
Master Boot Record	14
Extended Boot Record	15
Administración de usuarios y grupos.....	16
Archivo de usuarios.....	16
Login.....	16
Logout	17
Mkgrp.....	17
rmgrp.....	18
mkusr	18
rmusr.....	19
Usuario root	19
Administración de carpetas, archivos y permisos	20
chmod	20
mkfile	21
cat.....	22
rm.....	23
edit	24
ren	25
mkdir	25

cp.....	26
mv	27
find	29
chown.....	30
chgrp	31
Enlace simbólico y Enlace duro	31
Unlink	33
Convert EXT2 to EXT3.....	34
Pérdida y Recuperación del Sistema de Archivos EXT3	34
Recovery File System	34
Simulate System loss.....	35
Sistema de Archivos EXT2/EXT3.....	35
ESTRUCTURA DE SISTEMA DE ARCHIVOS EXT2.....	35
ESTRUCTURA DE SISTEMA DE ARCHIVOS EXT3.....	36
Súper Bloque	36
Journaling.....	37
Bitmap de Inodos	38
Bitmap de Bloques	38
Tabla de inodos.....	38
Bloque de Carpetas.....	40
Bloque de Archivos	40
Bloque de Apuntadores	41
Limitaciones	41
Otras operaciones.....	42
Reportes.....	43
rep.....	43
mbr.....	44
disk	46
Reporte de Journaling.....	46
inode	47
Block.....	47
bm_inode.....	48
bm_block.....	48

tree.....	49
sb.....	50
file	51
Ls+l	51
Ls+i	51
Scripts.....	52
exec	54
Entrega	55
FASE 1.....	55
FASE 2.....	55

Comandos

La aplicación será totalmente en consola, a excepción de los reportes en Graphviz. Esta no tendrá menús sino que se utilizarán comandos. No distinguirá entre mayúsculas y minúsculas. Hay parámetros obligatorios y opcionales que se identificarán al inicio con un símbolo (- obligatorios) y (+opcionales). Solo se puede colocar un comando por línea. **No se podrá utilizar alguna herramienta para interpretar estos comandos** (flex, bison u otros). Se recomienda hacer un autómata sencillo para interpretar los comandos.

Si se utiliza un parámetro que no está especificado en este documento, debe mostrar un mensaje de error. Se utilizarán espacios en blanco para separar cada parámetro. **Los parámetros pueden venir en cualquier orden.**

Si un comando necesita más de una línea se utilizará \ al final de la línea para indicar que continúa en la siguiente línea. También se podrán ejecutar archivos de scripts con estos comandos. Se podrá comentar cada comando Los comentarios de estos scripts empezarán con #. Estos comandos se explicarán en detalle a continuación.

Administración de Discos

Estos comandos permitirán crear archivos que simularán discos duros en los que se podrá formatear más adelante con el sistema de archivos ext2/ext3. Estos comandos están disponibles desde que se inicia el programa. Los comandos son:

mkdisk

Este comando creará un archivo binario que simulará un disco duro, estos archivos binarios tendrán la extensión dsk y su contenido al inicio será \0. Deberá ocupar físicamente el tamaño indicado por los parámetros, (no importa que el sistema operativo no muestre el tamaño exacto). Recibirá el nombre del archivo que simulará el disco duro y tendrá los siguientes parámetros:

Parámetro	Categoría	Descripción
-size	Obligatorio	Este parámetro recibirá un número que indicará el tamaño del disco a crear. Debe ser positivo y mayor que cero, si no se mostrará un error.
+unit	Opcional	Este parámetro recibirá una letra que indicará las unidades que utilizará el parámetro size. Podrá tener los siguientes valores: k que indicará que se utilizarán Kilobytes (1024 bytes) m en el que se utilizarán Megabytes (1024 * 1024 bytes) Este parámetro es opcional, si no se encuentra se creará un disco con tamaño en Megabytes. Si se utiliza otro valor debe mostrarse un mensaje de error.
-path	Obligatorio	Este parámetro será la ruta en el que se creará el archivo que representará el disco duro. Si las carpetas de la ruta no existen deberán crearse.
-name	Obligatorio	Este parámetro será el nombre del disco con extensión dsk. Si no contiene la extensión dsk debe mostrar un mensaje de error.

Ejemplos:

#Crea un disco de 3000 Kb en la carpeta home

```
Mkdisk -Size::3000 +unit::K -path::"/home/user/"  
  \ -name::"Disco1.dsk"
```

#Se crearán carpetas si no existen

```
mkdisk -size::5 +unit::M -path::"/home/mis discos/"  
  \ -name::"Disco3.dsk"
```

#Crearé un disco de 10 Mb ya que no hay parámetro unit

```
Mkdisk -size::10 -path::"/home/mis discos/"  
  \ -name::"Disco4.dsk"
```

rmdisk

Este parámetro elimina un archivo que representa a un disco duro mostrando un mensaje de confirmación para eliminar. Tendrá los siguientes parámetros:

Parámetro	Categoría	Descripción
-path	Obligatorio	Este parámetro será la ruta en el que se eliminará el archivo que representará el disco duro. Si el archivo no existe, debe mostrar un mensaje de error.

Ejemplos:

#Elimina Disco4.dsk

```
rmDisk -path::"/home/mis discos/Disco4.dsk"
```

fdisk

Este comando administra las particiones en el archivo que representa al disco duro. Siempre se utilizará el primer ajuste para buscar espacio dentro del disco y crear la partición.

IMPORTANTE: Este primer ajuste solo es para buscar espacio para crear la partición, es diferente del ajuste que utilizará cada partición para crear los archivos que se utilizará más adelante en el proyecto.

Deberá mostrar un error si no se pudo realizar la operación solicitada sobre la partición, especificando por qué razón no pudo crearse (Por espacio, por restricciones de particiones, etc.). No se considerará el caso de que se pongan parámetros incompatibles, por ejemplo en un mismo comando fdisk llamara a delete y add.

La estructura de cada disco se explicará más adelante. Tendrá los siguientes parámetros:

Parámetro	Categoría	Descripción
-size	Obligatorio al crear	Este parámetro recibirá un número que indicará el tamaño de la partición a crear. Debe ser positivo y mayor a cero, si no mostrará un mensaje de error.
+unit	Opcional	Este parámetro recibirá una letra que indicará las unidades que utilizará el parámetro size. Podrá tener los siguientes valores: B: que indicará que se utilizarán bytes K: que indicará que se utilizarán Kilobytes (1024 bytes) M: en el que se utilizarán Megabytes (1024 * 1024 bytes) Este parámetro es opcional, si no se encuentra se creará una partición en Kilobytes . Si se utiliza un valor diferente mostrará un mensaje de error.
-path	Obligatorio	Este parámetro será la ruta en la que se encuentra el disco en el que se creará la partición. Este archivo ya debe existir, si no se mostrará un error.
+type	Opcional	Indicará que tipo de partición se creará. Ya que es opcional, se tomará como primaria en caso de que no se indique. Podrá tener los siguientes valores: P: en este caso se creará una partición primaria.

		<p>E: en este caso se creará una partición extendida. L: Con este valor se creará una partición lógica.</p> <p>Nota: Las particiones lógicas sólo pueden estar dentro de la extendida sin sobrepasar su tamaño. Deberá tener en cuenta las restricciones de teoría de particiones: La suma de primarias y extendidas debe ser como máximo 4. Solo puede haber una partición extendida por disco. No se puede crear una partición lógica si no hay una extendida.</p> <p>Si se utiliza otro valor diferente a los anteriores deberá mostrar un mensaje de error.</p>
+fit	Opcional	<p>Indicará el ajuste que utilizará la partición para asignar espacio. Podrá tener los siguientes valores:</p> <p>BF: Indicará el mejor ajuste Best Fit FF: Utilizará el primer ajuste First Fit WF: Utilizará el peor ajuste Worst Fit</p> <p>Ya que es opcional, se tomará el peor ajuste si no está especificado en el comando. Si se utiliza otro valor que no sea alguno de los anteriores mostrará un mensaje de error.</p>
+delete	Opcional	<p>Este parámetro indica que se eliminará una partición. Este parámetro se utiliza junto con -name y -path. Se deberá mostrar un mensaje que permita confirmar la eliminación de dicha partición.</p> <p>Si la partición no existe deberá mostrar error. Si se elimina la partición extendida, deben eliminarse las particiones lógicas que tenga adentro.</p> <p>Recibirá los siguientes valores:</p> <p>Fast: Esta opción marca como vacío el espacio en la tabla de particiones. Full: Esta opción además marcar como vacío el espacio en la tabla de particiones, rellena el espacio con el carácter \0. Si se utiliza otro valor diferente, mostrará un mensaje de error.</p>
-name	Obligatorio	<p>Indicará el nombre de la partición. El nombre no debe repetirse dentro de las particiones de cada disco. Si se va a eliminar, la partición ya debe existir, si no existe debe mostrar un mensaje de</p>

		error.
+add	Opcional	<p>Este parámetro se utilizará para agregar o quitar espacio de la partición. Puede ser positivo o negativo. Tomará el parámetro +units para las unidades a agregar o eliminar.</p> <p>En el caso de agregar espacio, deberá comprobar que exista espacio libre después de la partición. En el caso de quitar espacio se debe comprobar que quede espacio en la partición (no espacio negativo).</p>

Ejemplos:

#Crea una partición primaria llamada Particion1 de 300 kb

#con el peor ajuste en el disco Disco1.dsk

```
fdisk -Size::300 -path::"/home/Disco1.dsk" -name::"Particion1"
```

#Crea una partición extendida dentro de Disco2 de 300 kb

#Tiene el peor ajuste

```
fdisk +type::E -path::"/home/Disco2.dsk" +Unit::K \
-name::"Particion2" -size::300
```

#Crea una partición lógica con el mejor ajuste, llamada Particion3,

#de 1 Mb en el Disco3

```
fdisk -size::1 +type::L +unit::M +fit::BF \
-path::"/mis discos/Disco3.dsk" name::"Particion3"
```

#Intenta crear una partición extendida dentro de Disco2 de 200 kb

#Debería mostrar error ya que ya existe una partición extendida

#dentro de Disco2

```
fdisk +type::E -path::"/home/Disco2.dsk" -name::"Part3" \
+Unit::K -size::200
```

#Elimina de forma rápida una partición llamada Particion1 fdisk +delete::fast -

```
name::"Particion1" -path::"/home/Disco1.dsk"
```

#Elimina de forma completa una partición llamada Particion1 fdisk -name::"Particion1"

```
+delete::full \ -path::"/home/Disco1.dsk"
```

#Agrega 1 Mb a la partición Particion4 del Disco4.dsk

#Se debe validar que haya espacio libre después de la partición fdisk +add::1 +unit::M -

```
path::"/home/mis discos/Disco4.dsk" \
-name::"Particion 4"
```

mount

Este comando montará una partición del disco en el sistema. Cada partición se identificará por un id que tendrá la siguiente estructura: VD + LETRA + NUMERO. Por ejemplo: vda1, vda2, vdb1, vdc1... La letra será la misma para particiones en el mismo disco y el número diferente para particiones en el mismo disco.

Parámetro	Categoría	Descripción
-path	Obligatorio	Este parámetro será la ruta en la que se encuentra el disco que se montará en el sistema. Este archivo ya debe existir.
-name	Obligatorio	Indica el nombre de la partición a cargar. Si no existe debe mostrar error.

Ejemplos:

#Monta las particiones de Disco1.dsk

```
mount -path::"/home/Disco1.dsk" -name::"Part1" #id::vda1
```

```
mount -path::"/home/Disco2.dsk" -name::"Part1" #id::vdb1
```

```
mount -path::"/home/Disco3.dsk" -name::"Part2" #id::vdc1
```

```
mount -path::"/home/Disco1.dsk" -name::"Part2" #id::vda2
```

#Si se coloca el comando mount sin parametros mostrará en la consola las particiones montadas.

```
#id::vda1 -path::"/home/Disco1.dsk" -name::"Part1"
```

```
#id::vdb1 -path::"/home/Disco2.dsk" -name::"Part1"
```

```
#id::vdc1 -path::"/home/Disco3.dsk" -name::"Part2"
```

```
#id::vda2 -path::"/home/Disco1.dsk" -name::"Part2"
```

umount

Desmonta una partición del sistema. Se utilizará el id que se le asignó a la partición al momento de cargarla. Recibirá los siguientes parámetros.

Parámetro	Categoría	Descripción
-idn	Obligatorio	Especifica una lista de id de las particiones que se desmontará. Si no existe algún id deberá mostrar un error.

Ejemplos:

#Desmonta la partición con id vda1 (En Disco1.dsk) `umount -id1::vda1`

#Si no existe, se debe mostrar error `umount -id1::vdx1`

#Desmonta una lista de particiones. `umount -id1::vda1 -id2::vdb2 -id3::vdc1`

disk free (df)

Este comando muestra la información del nombre del dispositivo, el total de bloques, el espacio total del disco, el espacio de disco utilizado, el espacio disponible en el disco. Si alguno de los parámetros no aparece entonces se visualizará el resultado en bytes.

Parámetro	Categoría	Descripción
+k	Opcional	Este parámetro será para visualizar toda la información del sistema de archivos y el uso de bloques de 1024 bytes.
+m	Opcional	Este parámetro será para visualizar toda la información del sistema de archivos y el uso de bloques de 1024 * 1024 bytes
+h	Opcional	Este parámetro será para mostrar los tamaños en formatos legibles por humanos mediante el uso de +h (imprime los resultados en formato legible por el hombre (por ejemplo, 1K 2M 3G)).
+i	Opcional	Con el parámetro +i mostrará la información del número de inodos utilizados y su porcentaje en el sistema de archivos.

Ejemplos:

```
#Filesystem Type      1K-blocks  Used    Available  Use% id
# /home/usr  Ext3      78361192 23186116 51130312  32% vda1
# /proc      Ext2         0      0         0      -    -
# /          Ext3     24797380 22273432 1243972   95% vdb1
Df +k
```

```
#Filesystem Type  blocks  Used    Available  Use% id
#/home/user Ext2   75G    23G     49G     32% vda1
#/proc      Ext3     0      0        0        -    -
# /         Ext3   24M    22M     1.2M     95% vdb1
Df +h
```

```
#Filesystem Type      Inodes   IUsed   IFree   IUse% id
#/home/user  Ext2    29G    25G    2.6G    91% vda1
#/proc       Ext3     0      0      0        -    -
# /          Ext3   24M    22M    1.2M    95% vdb1
Df +i +h
```

disk used (du)

Se usa para estimar el uso de espacio en disco duro de un archivo, un directorio en particular o de archivos en un sistema de archivos.

Parámetro	Categoría	Descripción
+n	Opcional	Especifica una lista de n directorios o archivos más pesados en nuestro sistema de archivos. Por defecto el valor es 1.
-h	Obligatorio	Especifica la cantidad en M, G o como considere más entendible.
-path	Opcional	Especifica el tamaño actual de un directorio o archivo en particular, si no se muestra este parámetro por defecto se tomará a evaluar todo el sistema de archivos.

Ejemplos:

Du -h -n::5

```
#215G  Videos
#171G  Linux
#68G   Documentos
#50G   Música
#28K   mageia-2013.txt
```

Du -h +n::3 +path::"/home/usr"

```
#Este mostrará los 3 primeros directorios o archivos que ocupen más espacio del directorio
# /home/usr
#215G  Videos
#171G  Linux
#68M   hola.txt
```

DU -H

```
#este mostrará el directorio o archivo que ocupe más espacio en el sistema de archivo.
#215G Videos
```

mkfs

Este comando realiza un formateo completo de la partición, se dará formato con ext2/ext3. También creará un archivo en la raíz llamado users.txt que tendrá los usuarios y contraseñas del sistema de archivos. La estructura de este archivo se explicará más adelante.

Parámetro	Categoría	Descripción
-id	Obligatorio	Indicará el id que se generó con el comando mount de la fase anterior. Si no existe mostrará error. Se utilizará para saber la partición y el disco que se utilizará para hacer el sistema de archivos.
+type	Opcional	Indicará que tipo de formato se realizará. Ya que es opcional, se tomará como un formato completo si no se especifica esta opción. Podrá tener los siguientes valores: Fast: en este caso se realizará un formato rápido. Full: en este caso se realizará un formato completo. La diferencia entre estos dos tipos se explicará más adelante.
+add	Opcional	Este parámetro se utilizará para aumentar o reducir el tamaño del sistema de archivos, el valor puede ser positivo o negativo. En caso de que sea negativo se debe validar que no se elimine información dentro del sistema ext2/ext3 al momento de reducir el sistema de archivos. En el caso de que sea positivo se debe validar que no sobrepase el tamaño de la partición.
+unit	Opcional	Este parámetro recibirá una letra que indicará las unidades que utilizará el parámetro +add . Podrá tener los siguientes valores: B: que indicará que se utilizarán bytes K: que indicará que se utilizarán Kilobytes (1024 bytes) M: en el que se utilizarán Megabytes (1024 * 1024 bytes) Este parámetro es opcional, si no se encuentra se utilizarán Kilobytes .
+FS	Opcional	Indica el sistema de archivos a formatear ext2 / ext3. Por defecto será ext3. Y los valores serán. 2fs o 3fs

Ejemplos:

#Realiza un formateo rápido con ext3 de la "Particion 1" del #Disco1.dsk

```
mkfs -id::vda1 -type::fast
```

#Realiza un formateo completo con ext2 de Particion2 en Disco2.dsk

```
mkfs -id::vdb1 +FS::2fs
```

#Agrega 500 Kb a Particion3 en Disco3.dsk

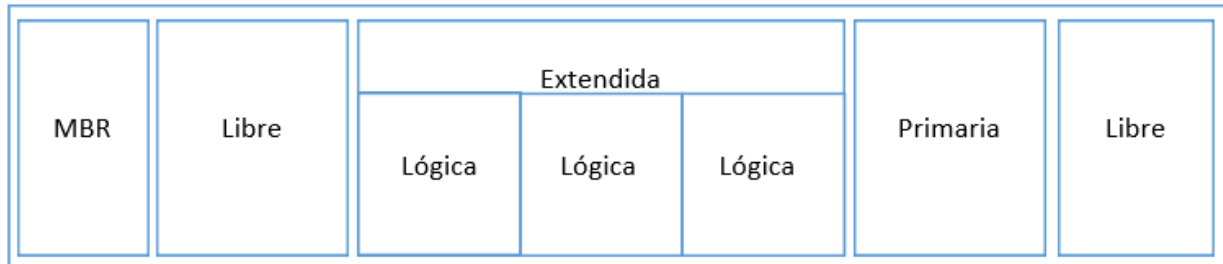
```
mkfs -add::500 -id::vdc1
```

#Agrega 1 Mb a Particion3 en Disco3.dsk

```
Mkfs -add::1 -id::vdc1 /  
-unit::M
```

Discos

Los discos serán archivos binarios que tendrán información del MBR, y espacio con particiones o bien, espacio sin utilizar. La siguiente figura es un ejemplo de los bloques en un disco con particiones en el que ya se ha eliminado una partición:



Master Boot Record

Cuando se crea una partición debe utilizarse el primer ajuste para crearla. En la figura anterior debería utilizarse el primer bloque libre para crear una nueva partición. El MBR tendrá los siguientes campos:

Nombre	Tipo	Descripción
mbr_tamaño	int	Tamaño total del disco en bytes
mbr_fecha_creacion	time	Fecha y hora de creación del disco
mbr_disk_signature	int	Número random, que identificará de forma única a cada disco
mbr_partition_1	partition	Estructura con información de la partición 1
mbr_partition_2	partition	Estructura con información de la partición 2
mbr_partition_3	partition	Estructura con información de la partición 3
mbr_partition_4	partition	Estructura con información de la partición 4

El contenido de la estructura partition será el siguiente:

Nombre	Tipo	Descripción
part_status	Char	Indica si la partición está activa o no
part_type	Char	Indica el tipo de partición, primaria o extendida. Tendrá los valores P o E
part_fit	Char	Tipo de ajuste de la partición. Tendrá los valores BF (Best), FF (First) o WF (worst)

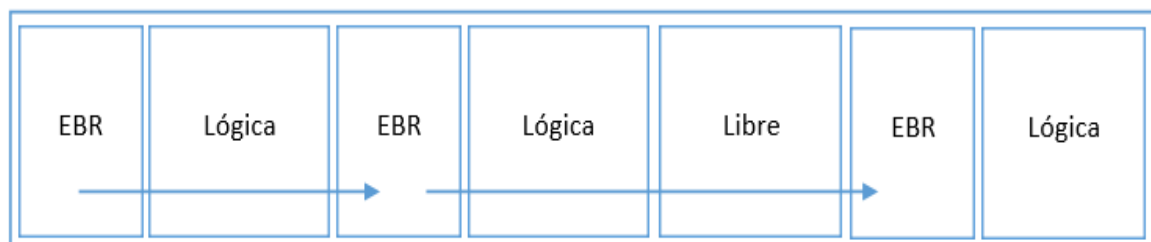
part_start	Int	Indica en qué byte del disco inicia la partición
part_size	Int	Contiene el tamaño total de la partición en bytes.
part_name	char[16]	Nombre de la partición

Extended Boot Record

Las particiones extendidas tendrán una estructura diferente. Se utilizará una estructura llamada EBR (Extended Boot Record) en forma de lista enlazada, que será como la siguiente:

Nombre	Tipo	Descripción
part_status	Char	Indica si la partición está activa o no
part_fit	Char	Tipo de ajuste de la partición. Tendrá los valores BF (Best), FF (First) o WF (worst)
part_start	Int	Indica en que byte del disco inicia la partición
part_size	Int	Contiene el tamaño total de la partición en bytes.
part_next	Int	Byte en el que está el próximo EBR. -1 si no hay siguiente
part_name	char[16]	Nombre de la partición

La estructura lógica de la partición extendida será como la siguiente:



EBR inicial siempre debe existir, aunque se elimine la primera partición.

Para crear el archivo del disco se recomienda utilizar un char[1024] como buffer para crear el archivo, si se utiliza un char[1] normalmente se tarda demasiado al momento de crear el archivo.

Administración de usuarios y grupos

Archivo de usuarios

Este archivo será un archivo de texto, llamado users.txt guardado en el sistema ext2/ext3 de la raíz de cada partición. Existirán dos tipos de registros, unos para grupos y otros para usuarios. Un id 0 significa que el usuario o grupo está eliminado, el id de grupo o de usuario irá aumentando según se vayan creando usuarios o grupos. Tendrá la siguiente estructura:

GID, Tipo, Grupo

UID, Tipo, Grupo, Usuario, Contraseña

El estado ocupará una letra, el tipo otra, el grupo ocupará como máximo 10 letras al igual que el usuario y la contraseña.

Al inicio existirá un grupo llamado root, un usuario root y una contraseña (#carné) para el usuario root. El archivo al inicio debería ser como el siguiente:

1, G, root \n

1, U, root , root , 201212838 \n

Este archivo se podrá modificar con comandos que se explicarán más adelante.

Login

Este comando se utiliza para iniciar sesión en el sistema. Se buscará el usuario únicamente en el archivo users.txt de la primera partición cargada, vda1 o el que tenga el id menor. No se puede iniciar otra sesión sin haber hecho un logout antes, si no, debe mostrar un mensaje de error indicando que debe cerrar sesión antes. Recibirá los siguientes parámetros:

Parámetro	Categoría	Descripción
-usr	Obligatorio	Especifica el nombre del usuario que iniciará sesión. Si no se encuentra en la primera partición mostrará un mensaje indicando que el usuario no existe. En este caso si distinguirá mayúsculas de minúsculas.
-pwd	Obligatorio	Indicará la contraseña del usuario, si no coincide debe mostrar un mensaje de autenticación fallida. Distinguirá entre mayúsculas y minúsculas.

Ejemplos:

#Se loguea en el sistema como usuario root

login -usr::root -pwd::201212838

login -usr::"mi usuario" -pwd::"mi pwd"

Logout

Este comando se utiliza para cerrar sesión. Debe haber una sesión activa anteriormente para poder utilizarlo, si no, debe mostrar un mensaje de error. Este comando no recibe parámetros.

Ejemplo:

#Termina la sesión del usuario

Logout

#Si se vuelve a ejecutar deberá mostrar un error

#ya que no hay sesión actualmente

Logout

Todos los siguientes comandos que se explicarán de aquí en adelante, **necesitan que exista una sesión en el sistema**. Si no, debe mostrar un mensaje de error indicando que necesita iniciar sesión.

Mkgrp

Este comando creará un grupo para los usuarios de la partición y se guardará en el archivo users.txt de la partición, este comando solo lo puede utilizar el usuario root. Si otro usuario lo intenta ejecutar, deberá mostrar un mensaje de error, si el grupo a ingresar ya existe deberá mostrar un mensaje de error. Recibirá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se creará el grupo. Si no existe, debe mostrar un error.
-name	Obligatorio	Indicará el nombre que tendrá el grupo

Ejemplo:

#Crea el grupo usuarios en la partición vda1

mkgrp -id::vda1 -name::"usuarios"

#Debe mostrar mensaje de error ya que el grupo ya existe

mkgrp -id::vda1 -name::"usuarios"

El archivo users.txt debería quedar como el siguiente:

```
1, G, Root      \n
1, U, root      , root  , 123  \n
2, G, usuarios \n
```

rmgrp

Este comando eliminará un grupo para los usuarios de la partición. Solo lo puede utilizar el usuario root, si lo utiliza alguien más debe mostrar un error. Recibirá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en el que se eliminará el grupo. Si no existe, debe mostrar un error.
-name	Obligatorio	Indicará el nombre del grupo a eliminar. Si el grupo no se encuentra dentro de la partición debe mostrar un error.

Ejemplo:

#Elimina el grupo de usuarios en la partición vda1

```
rmgrp -id::vda1 -name::"usuarios"
```

#Debe mostrar mensaje de error ya que el grupo no existe porque ya fue eliminado

```
rmgrp -id::vda1 -name::"usuarios"
```

El archivo users.txt debería quedar como el siguiente:

```
1, G, Root      \n
1, U, root      , root  , 201212838  \n
0, G, usuarios \n
```

mkusr

Este comando crea un usuario en la partición. Solo lo puede ejecutar el usuario root, si lo utiliza otro usuario deberá mostrar un error. Recibirá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en el que se creará el usuario. Si no existe, debe mostrar un error.
-usr	Obligatorio	Indicará el nombre del usuario a crear, si ya existe, deberá mostrar un error indicando que ya existe el usuario. Máximo: 10 caracteres.
-pwd	Obligatorio	Indicará la contraseña del usuario. Máximo: 10 caracteres.
-grp	Obligatorio	Indicará el grupo al que pertenece el usuario. Debe de existir en la partición en la que se está creando el usuario, si no debe mostrar un mensaje de error. Máximo: 10 caracteres.

Ejemplo:

#Crea el grupo usuarios en la partición vda1

Mkusr -id::vda1 -name::"user1" -grp::usuarios -pwd::usuario

El archivo users.txt debería quedar así:

```
1, G, Root      \n
1, U, root      , root  , 201212838 \n
2, G, usuarios \n
2, U, usuarios , user1  , usuario \n
```

rmusr

Este comando elimina un usuario en la partición. Solo lo puede ejecutar el usuario root, si lo utiliza otro usuario deberá mostrar un error. Recibirá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se eliminará el usuario. Si no existe, debe mostrar un error.
-usr	Obligatorio	Indicará el nombre del usuario a eliminar, si no existe, deberá mostrar un error indicando que el usuario no existe.

Ejemplo:

#Elimina el usuario user1 de la partición

rmusr -id::vda1 -usr::user1

El archivo users.txt debería quedar así:

```
1, G, Root      \n
1, U, root      , root  , 123    \n
2, G, usuarios \n
0, U, usuarios , user1  , usuario \n
```

Usuario root

Este usuario es especial y no importando que permisos tenga el archivo o carpeta, el siempre tendrá los permisos 777 sobre cualquier archivo o carpeta (Esto se explica en detalle posteriormente). Podrá mover, copiar, eliminar, crear, etc. Todos los archivos o carpetas que desee. No se le negará ninguna operación por permisos, ya que él los tiene todos. Los permisos únicamente se pueden cambiar con chmod que se explicará posteriormente. Se debe tomar en cuenta en qué categoría está el usuario, si es el propietario, si pertenece al mismo grupo en que está el propietario o si es otro usuario que no pertenece al grupo del propietario. En base a esta comprobación, el usuario puede estar en tres distintas categorías: Propietario (U), Grupo (G) u Otro (O). Dependiendo de estas categorías se determinan los permisos hacia el archivo o carpeta.

Administración de carpetas, archivos y permisos

Estos comandos permitirán crear archivos y carpetas, así como editarlos, copiarlos, moverlos y eliminarlos. Los permisos serán para el usuario propietario del archivo, para el grupo al que pertenece y para otros usuarios, así como en Linux.

chmod

Cambiará los permisos de uno o varios archivos o carpetas. Lo podrá utilizar el usuario root en **todos** los archivos o carpetas y también lo podrán utilizar otros usuarios, pero **solo** sobre sus **propios** archivos. Recibirá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se encuentra el archivo o carpeta a la que se le cambiarán los permisos.
-path	Obligatorio	Este parámetro será la ruta en la que se encuentra el archivo o carpeta a la que se le cambiarán los permisos.
-ugo	Obligatorio	Indica los permisos que tendrán los usuarios. Serán tres números, uno para el U usuario, el siguiente para el G Grupo al que pertenece el usuario y el último para O tros usuarios fuera del grupo. Cada número tendrá los valores desde el 0 al 7. Si el número está fuera de este rango se mostrará un error. A nivel de bits significan permisos para lectura, escritura y ejecución. Por ejemplo el número 5 (101) indica permisos para leer y ejecutar. El número 7 indica permisos para las tres operaciones anteriores. El número 0 indica que no tendrá permisos para utilizar el archivo.
+R	Opcional	Indica que el cambio será recursivo en el caso de carpetas. El cambio afectará a todos los archivos y carpetas en la que la ruta contenga la carpeta especificada por el parámetro -path y que sean propiedad del usuario actual.

Ejemplos:

```
#Cambia los permisos de la carpeta home recursivamente
#Todos los archivos o carpetas que tengan /home cambiarán
#Por ejemplo si existiera /home/user/docs/a.txt
#Cambiaría los permisos de las tres carpetas y del archivo
chmod -id::vda1 -path::"/home" +R -ugo::764
```

```
#Cambia los permisos de la carpeta home
#Se debe comprobar que la carpeta home pertenezca al usuario #actual, si no deberá mostrar un mensaje de error.
chmod -id::vda1 -path::"/home" -ugo::777
```

mkfile

Este comando permitirá crear un archivo, el propietario será el usuario que actualmente ha iniciado sesión. Tendrá los permisos 664. El usuario deberá tener el permiso de escritura en la carpeta padre, si no debe mostrar un error. Tendrá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se creará el archivo. Si no existe debe mostrar error.
-path	Obligatorio	Este parámetro será la ruta del archivo que se creará. Si lleva espacios en blanco deberá encerrarse entre comillas. Si ya existe debe sobre escribir el archivo. Si no existen las carpetas padres, debe mostrar error, a menos que se utilice el parámetro -p, que se explica posteriormente.
+p	Opcional	Si se utiliza este parámetro y las carpetas especificadas por el parámetro -path no existen, entonces deben crearse las carpetas padres. Si ya existen, no deberá crear las carpetas. No recibirá ningún valor, si lo recibe debe mostrar error.
+size	Opcional	Este parámetro indicará el tamaño en bytes del archivo, el contenido serán números del 0 al 9 cuantas veces sea necesario. Si no se utiliza este parámetro, el tamaño será 0 bytes. Si es negativo debe mostrar error.
+cont	Opcional	Indicará un archivo en el disco duro de la computadora que tendrá el contenido del archivo. Se utilizará para cargar contenido en el archivo. La ruta ingresada debe existir, si no mostrará un mensaje de error.

Ejemplos:

```
#Crea el archivo a.txt  
#Si no existen las carpetas home user o docs se crean  
#El tamaño del archivo es de 15 bytes  
#El contenido sería: 012345678901234
```

```
mkFile +SIZE::15 -id::vdb1 -PatH::"/home/user/docs/a.txt" -p
```

#Crea "archivo 1.txt" la carpeta "mis documentos" ya debe existir

#el tamaño es de 0 bytes

```
mkfile -id::vda1 -path::"/home/mis documentos/archivo 1.txt"
```

#Crea el archivo b.txt

#El contenido del archivo será el mismo que el archivo b.txt

#que se encuentra en el disco duro de la computadora.

```
mkfile -id::vda1 -path::"/home/user/docs/b.txt" +p \  
+cont::"/home/Documents/b.txt"
```

cat

Este comando permitirá mostrar el contenido del archivo, si el usuario que actualmente está logueado tiene acceso al permiso de lectura. Tendrá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se leerá el archivo. Si no existe debe mostrar error.
-filen	Obligatorio	Permitirá Admitir como argumentos una lista de n ficheros que hay que enlazar. Estos se encadenarán en el mismo orden en el cual fueron especificados. Si no existe el archivo o no tiene permiso de lectura, debe mostrarse un mensaje de error.

Ejemplos:

#Lee el archivo a.txt En la terminal debería mostrar el contenido, en #este ejemplo 01234567890123

```
Cat -file1::"/home/user/docs/a.txt" -Id::vdb1
```

#enlazara los archivos a.txt (datos archivo a) b.txt (01234567890123) #c.txt (0123) y debería mostrar el contenido siguiente, cada archivo #va separado por salto de línea

datos archivo a

01234567890123

0123

```
Cat -file1::"/home/a.txt" -Id::vdb1 -file2::"/home/b.txt" \ -file3::"/home/c.txt"
```

rm

Este comando permitirá eliminar un archivo o carpeta y todo su contenido, si el usuario que actualmente está logueado tiene acceso al permiso de escritura sobre el archivo y en el caso de carpetas, eliminará todos los archivos o subcarpetas en los que el usuario tenga permiso de escritura. Si no pudo eliminar un archivo o subcarpeta dentro de la carpeta por permisos, no deberá eliminar la(s) carpetas padre. Tendrá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se eliminará el archivo o carpeta. Si no existe debe mostrar error.
-path	Obligatorio	Este parámetro será la ruta del archivo o carpeta que se eliminará. Deberá encerrarse entre comillas. Si no existe el archivo o no tiene permisos de escritura en la carpeta o en el archivo, debe mostrarse un mensaje de error. Si no pudo eliminar algún archivo o carpeta no deberá eliminar los padres.
+rf	Opcional	Permite hacer borrados recursivos. Con esta opción se pueden borrar sin importar los permisos siempre y cuando sea el usuario root.

Ejemplos:

#Elimina el archivo a.txt, b.txt muestra error si no tiene permiso

```
rm -Path::"/home/user/docs/a.txt" -Id::vdb1
```

```
rm -Path::"/home/user/docs/b.txt" -Id::vdb1 #Error por permisos
```

#Elimina la carpeta user y todo su contenido (docs, a.txt)

#Si el usuario no tuviera permiso de escritura sobre b.txt

#No debería eliminar las carpetas padre docs ni user, solo a.txt

```
rm -Path::"/home/user" -Id::vdb1
```

#Elimina la carpeta user y todo su contenido (docs, a.txt) sin importar los permisos (logueado como usuario root)

```
Rm -Id::vdb1 -path::"/home/user" +RF
```

edit

Este comando permitirá editar el contenido de un archivo para que ocupe un tamaño específico, o bien, para asignarle otro contenido. Funcionará si el usuario que actualmente esta logueado tiene acceso al permiso de lectura y escritura sobre el archivo, si no debe mostrar error. Tendrá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se modificará el archivo. Si no existe debe mostrar error.
-path	Obligatorio	Este parámetro será la ruta del archivo que se modificará. Deberá encerrarse entre comillas. Si no existe, debe mostrar un mensaje de error.
+size	Opcional*	Este parámetro indicará el nuevo tamaño en bytes del archivo, el contenido serán números del 0 al 9 cuantas veces sea necesario. Si es negativo debe mostrar error.
+cont	Opcional*	Indicará un archivo en el disco duro de la computadora que tendrá el nuevo contenido del archivo. Se utilizará para cargar contenido en el archivo. La ruta ingresada debe existir, si no mostrará un mensaje de error.

(*) Se deberá ingresar por lo menos uno de los dos parámetros (size o cont). Si no hay ninguno de los dos, debe mostrar un mensaje de error.

Ejemplos:

```
#Modifica el archivo a.txt
#El tamaño del archivo es de 22 bytes
#El contenido sería: 0123456789012345678901
Edit +SIZE::22 -id::vdb1 -PatH::"/home/user/docs/a.txt"
```

```
#Modifica el archivo b.txt
#El contenido del archivo será el mismo que el archivo c.txt
#que se encuentra en el disco duro de la computadora.
edit -id::vda1 -path::"/home/user/docs/b.txt" \
+cont ::"/home/Documents/c.txt"
```

```
#Modifica nuevamente el archivo b.txt
edit -id::vda1 -path::"/home/user/docs/b.txt" \
+cont ::"/home/Documents/d.txt"
```


ren

Este comando permitirá cambiar el nombre de un archivo o carpeta, si el usuario actualmente logueado tiene permiso de escritura sobre el archivo o carpeta. Tendrá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se renombrará el archivo o carpeta. Si no existe debe mostrar error.
-path	Obligatorio	Este parámetro será la ruta del archivo o carpeta al que se le cambiará el nombre. Deberá encerrarse entre comillas. Si no existe el archivo o carpeta o no tiene permisos de escritura sobre la carpeta o archivo, debe mostrarse un mensaje de error.
-name	Obligatorio	Especificará el nuevo nombre del archivo, debe verificar que no exista un archivo con el mismo nombre, de ser así debe mostrar un mensaje de error. Deberá encerrarse entre comillas.

Ejemplos:

#Cambia el nombre del archivo a.txt a b1.txt

```
ren -Path::"/home/user/docs/a.txt" -Id::vdb1 -name::"b1.txt"
```

#Debera mostrar error ya que el archivo b1.txt ya existe

```
ren -Path::"/home/user/docs/c.txt" -Id::vdb1 -name::"b1.txt"
```

mkdir

Este comando es similar a mkfile, pero no crea archivos, sino carpetas. El propietario será el usuario que actualmente ha iniciado sesión. Tendrá los permisos 664. . El usuario deberá tener el permiso de escritura en la carpeta padre, si no debe mostrar un error. Tendrá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se creará la carpeta. Si no existe debe mostrar error.
-path	Obligatorio	Este parámetro será la ruta de la carpeta. Si no existen las carpetas padres, debe mostrar error, a menos que se utilice el parámetro +p, que se explica posteriormente.
+p	Opcional	Si se utiliza este parámetro y las carpetas padres en el parámetro path no existen, entonces deben crearse.

		Si ya existen, no realizará nada. No recibirá ningún valor, si lo recibe debe mostrar error.
--	--	--

Ejemplos:

#Crea la carpeta usac

#Si no existen las carpetas home user o docs se crean

Mkdir +P -id::vda1 -path::"/home/user/docs/usac"

#Crea la carpeta "archivos 2016"

#La carpeta padre ya debe existir

Mkdir -ID::vda1 -path::"/home/mis documentos/archivos 2016"

cp

Este comando permitirá realizar una copia del archivo o carpeta y todo su contenido hacia otro destino.

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que está el archivo o carpeta que se quiere copiar. Si no existe, debe mostrar un error.
-path	Obligatorio	Este parámetro será la ruta del archivo o carpeta que se desea copiar. Debe copiar todos los archivos y carpetas con todo su contenido, a los cuales tenga permiso de lectura. Si no tiene permiso de lectura, no realiza la copia únicamente de ese archivo o carpeta. Muestra un error si no existe la ruta.
-dest	Obligatorio	Este parámetro será la ruta de la carpeta a la que se copiará el archivo o carpeta. Debe tener permiso de escritura sobre la carpeta. Debe mostrar un mensaje de error si no tiene permisos para escribir o si la carpeta no existe.
-iddest	Obligatorio	Especifica el id de la partición en la que se copiará el archivo o carpeta. Si no existe debe mostrar un error.

Ejemplos:

```
#/  
#|_home      #664  
# |_user      #664  
# | |_documents #664  
# | |_a.txt    #664  
# | |_b.txt    #224  
# |_images    #664
```

#Copia documents a images

```
cp -id::vda2 -Path::"/home/user/documents" \  
-iddest::vda2 -dest::"/home/images"
```

#No copia b.txt por falta de permisos

```
#/  
#|_home      #664  
# |_user      #664  
# | |_documents #664  
# | |_a.txt    #664  
# | |_b.txt    #224  
# |_images    #664  
# |_documents #664  
# |_a.txt     #664
```

mv

Este comando moverá un archivo o carpeta y todo su contenido hacia otro destino. Si el origen y destino están dentro de la misma partición, solo cambiará las referencias, para que ya no tenga el padre origen sino, el padre destino, y que los padres de la carpeta o archivo ya no tengan como hijo a la carpeta o archivo que se movió. Solo se deberán verificar los permisos de escritura sobre la carpeta o archivo origen.

Si el origen y destino están en una partición diferente, funcionará como copiar y después eliminar. El usuario deberá tener el permiso de escritura en la carpeta padre destino, si no debe mostrar un error.

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que está el archivo o carpeta que se quiere mover. Si no existe, debe mostrar un error.

-path	Obligatorio	Este parámetro será la ruta del archivo o carpeta que se desea mover. Si no tiene permiso de escritura, no mueve el archivo o carpeta. Muestra error si no existe o no tiene permiso.
-dest	Obligatorio	Este parámetro será la ruta de carpeta a la que se moverá el archivo o carpeta. Debe tener permiso de escritura sobre la carpeta. Debe mostrar un mensaje de error si no tiene permiso de escritura o si la carpeta no existe.
-iddest	Obligatorio	Especifica el id de la partición en la que se moverá el archivo o carpeta. Si no existe debe mostrar un error.

Ejemplo:

```
#/
#|_home      #664
# |_user      #664
# | |_documents #664
# | |_a.txt    #664
# | |_b.txt    #224
# |_images    #664
```

#Mueve documents a images

```
mv -id::vda1 -iddest::vda1 -Path::"/home/user/documents" \
  -dest::"/home/images"
```

#Mueve b.txt, ya que solo se comprueban los permisos de documents

```
#/
#|_home      #664
# |_user      #664
# |_images    #664
# |_documents #664
# |_a.txt     #664
# |_b.txt     #224
```

find

Este comando permitirá realizar una búsqueda por el nombre del archivo o carpeta. Permitirá los siguientes caracteres especiales:

Carácter	Descripción
?	Un solo carácter
*	Uno o más caracteres

Recibirá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se buscará el archivo o carpeta. Si no existe, debe mostrar un error.
-path	Obligatorio	Este parámetro será la ruta de la carpeta en el que se inicia la búsqueda, deberá buscar en todo su contenido. Debe tener permisos de lectura en los archivos que buscará.
-name	Obligatorio	Indica el nombre del archivo o carpeta que se está buscando. Debe aceptar los caracteres especiales definidos anteriormente
+perm	Opcional	Encontrará archivos o carpetas con algún criterio de permisos. +perm::775
+user	Opcional	Buscará archivos o carpeta por un usuario en particular

El resultado debe mostrarse en forma de árbol, mostrando todos los archivos encontrados. `find *` mostrará toda la estructura del sistema de archivos a partir de la carpeta indicada ya que no hay filtro.

Ejemplos:

```
find -id::vda1 -Path::"/" -name::*
```

```
#Arbol actual
```

```
#/
```

```
#|_home      #664
```

```
# |_user      #664
```

```
# | |_a.txt    #664
```

```
# | |_b.txt    #420
```

```
# |_images    #664
```

```
#      |_a.jpg #664
```

```
#      |_abcd.jpg #664
```

```
#Busca los archivos que tengan una letra como nombre
#y cualquier extensión
find -id::vda1 -Path::"/home" -name::"?.*"
```

```
#El resultado del comando sería
#/
#|_home
# |_user
# |_a.txt
# |_b.txt
# |_images
# |_a.jpg
```

chown

Cambiará el propietario de uno o varios archivos o carpetas. Lo podrá utilizar el usuario root en todos los archivos o carpetas y también lo podrán utilizar otros usuarios, pero solo sobre sus propios archivos. Recibirá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición en la que se encuentra el archivo o carpeta a la que se le cambiará el propietario.
-path	Obligatorio	Este parámetro será la ruta en la que se encuentra el archivo o carpeta a la que se le cambiará el propietario. Si no existe la ruta deberá mostrar mensaje de error.
+R	Opcional	Indica que el cambio será recursivo en el caso de carpetas. El cambio afectará a todos los archivos y carpetas en la que la ruta contenga la carpeta especificada por el parámetro -path.
-usr	Obligatorio	Nombre del nuevo propietario del archivo o carpeta. Si no existe o está eliminado debe mostrar error.

Ejemplos:

```
#Cambia el propietario de la carpeta home recursivamente
chown -id::vda1 -path::"/home" +R -usr::user2
```

```
#Cambia los permisos de la carpeta home
chown -id::vda1 -path::"/home" -usr::user1
```

chgrp

Cambiará el grupo al que pertenece el usuario. Únicamente lo podrá utilizar el usuario root. Recibirá los siguientes parámetros:

Parámetro	Categoría	Descripción
-usr	Obligatorio	Especifica el nombre del usuario al que se le cambiará de grupo. Si no existe debe mostrar un error.
-grp	Obligatorio	Contendrá el nombre del nuevo grupo al que pertenece el usuario. Si no existe o está eliminado debe mostrar un error.

Ejemplos:

#Cambia el grupo del user2

chgrp -usr::user2 -grp::grupo1

#Cambia el grupo del user1

chgrp -usr::user1 -grp::grupo2

Enlace simbólico y Enlace duro

Un enlace simbólico permite dar a un fichero el nombre de otro, pero no enlaza el fichero con un inodo, es decir, en realidad lo que hacemos es enlazar directamente al nombre del fichero. Los enlaces simbólicos son ampliamente usados para las librerías compartidas.

Para comprenderlo mejor, un "enlace simbólico" no es más que una referencia (enlace) a un fichero que está situado en un lugar físico distinto.

Los enlaces duros lo que hacen es asociar dos o más ficheros compartiendo el mismo inodo. Esto hace que cada enlace duro es una copia exacta del resto de ficheros asociados, tanto de datos como de permisos, propietario, etc. Esto implica también que cuando se realicen cambios en uno de los enlaces o en el fichero este también se realizará en el resto de enlaces.

Parámetro	Categoría	Descripción
+s	Opcional	Específica que se realizará un enlace simbólico.
-source	Obligatorio	Es el archivo origen.
-link	Obligatorio	Es el archivo enlace a origen

Ejemplos:

#vamos a crear un enlace simbólico (parámetro -s) del fichero test:

ln +s -source::test.txt -link::enlace-a-test.txt

#Si listamos ambos veremos que el enlace tiene el carácter l que lo identifica como enlace #simbólico: ls -l (este reporte está explicado en la sección del comando REP)

```
# lrwxrwxrwx 1 alex alex 4 2011-04-27 18:59 enlace-a-test -> test
# -rw-r--r-- 1 alex alex 0 2011-04-27 18:58 test
```

#Para confirmar que el enlace simbólico tiene un inodo distinto usamos el reporte ls -li

l (este reporte está explicado en la sección del comando REP)

```
# 77212 lrwxrwxrwx 1 alex alex 4 2011-04-27 18:59 enlace-a-test -> test
# 73793 -rw-r--r-- 1 alex alex 0 2011-04-27 18:58 test
```

#vamos a crear un enlace duro del fichero test:

```
ln -source::"/home/test.txt" -link::"/home/user/enlace-duro-test.txt"
```

#Con el reporte ls -li (este reporte está explicado en la sección del comando REP)

```
#73793 -rw-r--r-- 2 alex alex 5 2011-04-27 19:09 enlace-duro-test
#73793 -rw-r--r-- 2 alex alex 5 2011-04-27 19:09 test
```

#En la primera columna verificamos que tienen el mismo número de inodo y en la tercera se #especifica cuando enlaces duros tiene el fichero. Si se hacen cambios en uno de ellos se #observa que también se hacen en el resto. Si por ejemplo cambiamos los permisos al fichero #test: chmod 0755 test

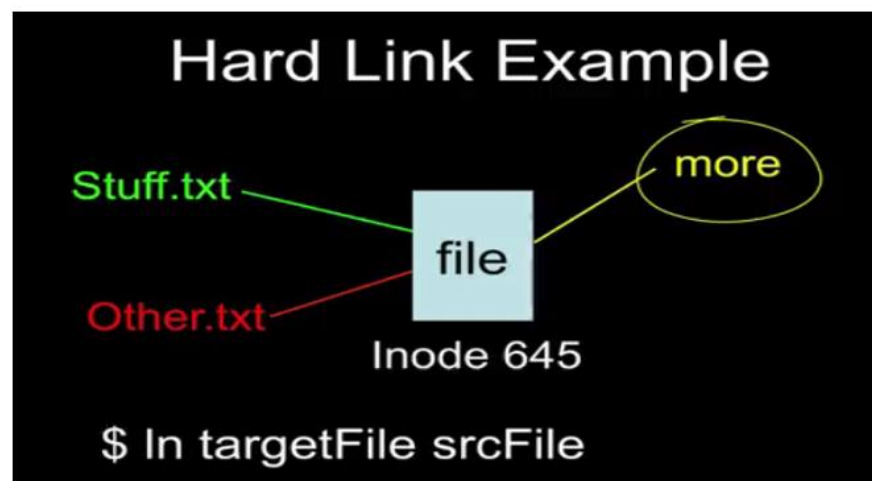
ls -li

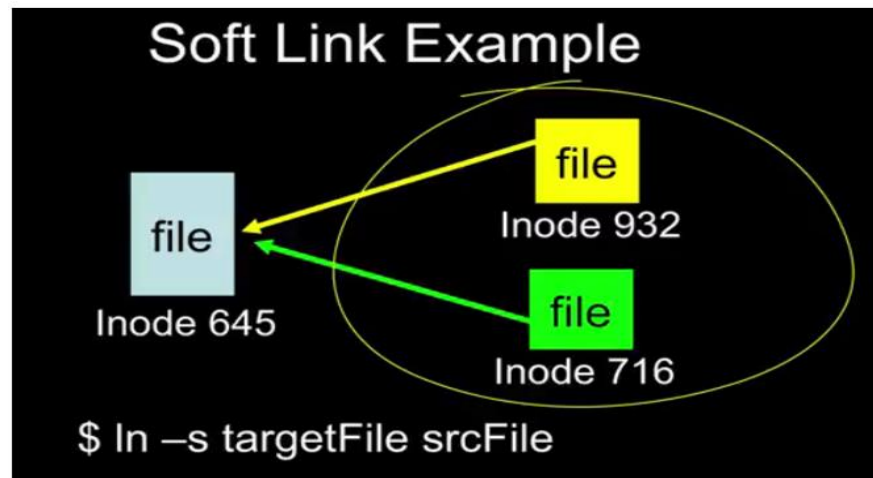
```
# 73793 -rwxr-xr-x 2 alex alex 5 2011-04-27 19:09 enlace-duro-test
```

```
# 73793 -rwxr-xr-x 2 alex alex 5 2011-04-27 19:09 test
```

Resumen:

- ✓ Los enlaces duros comparten el número de inodo, los simbólicos no.
- ✓ En los enlaces simbólicos si se borra el fichero, la información se pierde, en los duros no.
- ✓ Los enlaces duros son copias exactas del fichero mientras que los simbólicos son puros punteros o "accesos directos".





Unlink

Para borrar enlaces simbólicos o duros usaremos el comando Unlink. El procedimiento para eliminar enlaces es el mismo independientemente del tipo de enlace a eliminar.

Parámetro	Categoría	Descripción
-link	Obligatorio	Es el archivo enlace a origen

Ejemplo:

#Eliminar enlace simbólico

Unlink -link::"/home/user/enlace-a-test.txt"

Convert EXT2 to EXT3

Si actualmente la partición cuenta con un sistema de archivos ext2, es una buena idea activar journal y convertirlo en un sistema de archivos ext3. Tenemos que estar logueados como usuario root.

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica la partición que va hacer formateada con el sistema de archivos EXT3. Debe hacerse validaciones como verificar si la partición existe y si no ha sido ya formateada con el sistema de archivos EXT3.

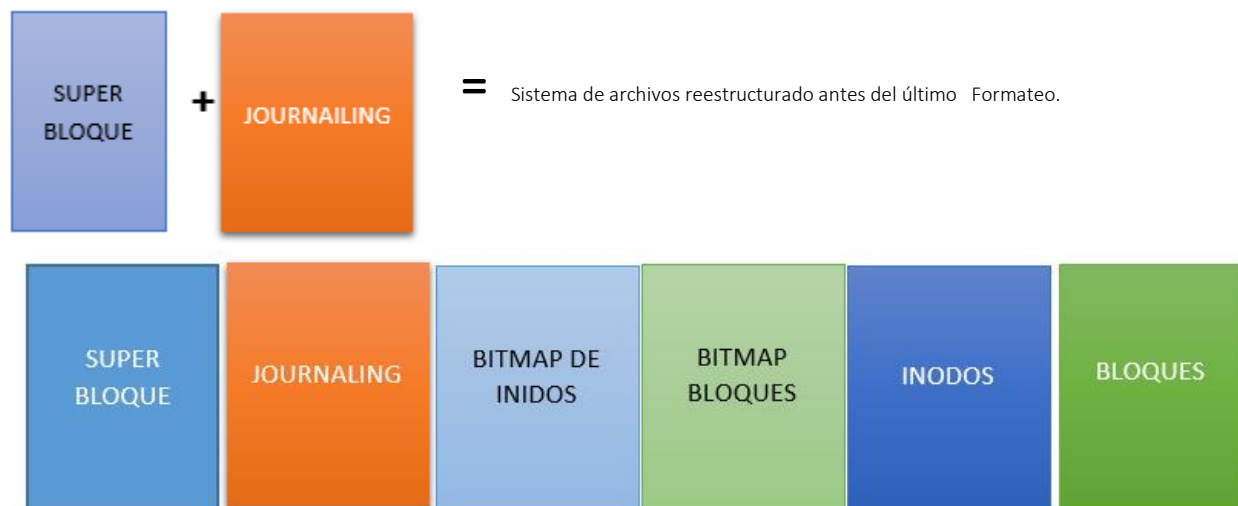
#Cambia la partición de ext2 a ext3, tiene que hacer las validaciones correspondientes para #saber si tiene espacio disponible para poder formatear con el sistema de archivos ext3. De lo #contrario debe mostrar error.

```
Tune2fs -id::vda1
```

Pérdida y Recuperación del Sistema de Archivos EXT3

Recovery File System

La recuperación del sistema se hará por medio del journaling y el superbloque. Se recuperará el sistema a un estado consistente antes del último formateo.



#Recuperando el sistema de archivos EXT3 de la partición 1

```
Recovery -id::vda1
```

Simulate System loss

Este formatea los siguientes bloques de datos para simular un fallo en el disco (una partición en especifica), una inconsistencia o pérdida de información. Se deberán limpiar los siguientes bloques con el carácter /0.

1. Bloque de bitmap de árbol virtual de directorio
2. Bloque de bitmap de Bloques
3. Inodos
4. Bloque de datos

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición a la que se le simulara la pérdida del sistema.

Ejemplo:

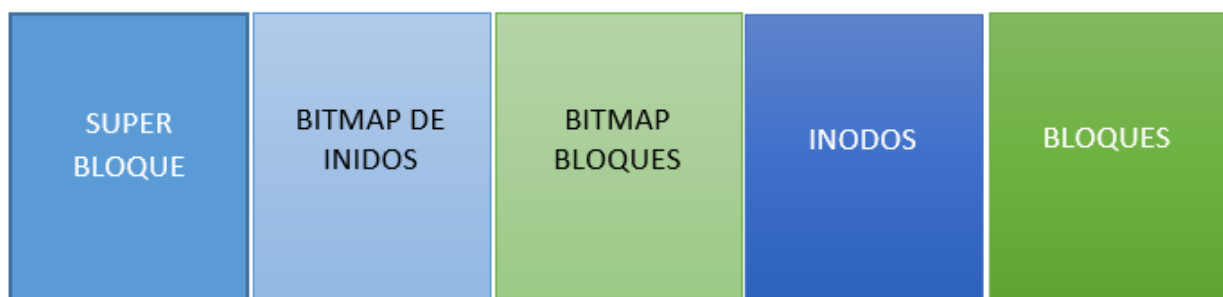
#Simulando la perdida del sistema de archivos EXT3 de la partición1

Loss -id::vda1

Sistema de Archivos EXT2/EXT3

A continuación se explicarán las estructuras del sistema de archivos EXT2/EXT3. Se deberán implementar las estructuras como se especifican a continuación. La estructura en bloques es la siguiente:

ESTRUCTURA DE SISTEMA DE ARCHIVOS EXT2



El número de bloques será el triple que el número de inodos. El número de inodos y bloques a crear se puede calcular despejando n de la primera ecuación y aplicando la función floor al resultado:

$$\text{tamaño_particion} = \text{sizeof}(\text{superblock}) + n + 3 * n + n * \text{sizeof}(\text{inodos}) + 3 * n * \text{Sizeof}(\text{block})$$

$$\text{numero_estructuras} = \text{floor}(n)$$

ESTRUCTURA DE SISTEMA DE ARCHIVOS EXT3



El número de bloques será el triple que el número de inodos. El número de Journaling, inodos y bloques a crear se puede calcular despejando n de la primera ecuación y aplicando la función floor al resultado:

$$\text{tamaño_particion} = \text{sizeof}(\text{superblock}) + n + n * \text{Sizeof}(\text{Journaling}) + 3 * n + n * \text{sizeof}(\text{inodos}) + 3 * n * \text{Sizeof}(\text{block})$$

$$\text{numero_estructuras} = \text{floor}(n)$$

Ya que el comando mkfs con el parámetro `-add` agrega o quita espacio de la partición, se deberán modificar estas estructuras sin eliminar información. Se calcula cuántos bloques se agregan o quitarán y se empieza moviendo de derecha a izquierda en el caso de agregar y en el caso de quitar bloques se empieza moviendo de izquierda a derecha para evitar sobrescribir información.

Súper Bloque

Contiene información sobre la configuración del sistema de archivos. Tendrá los siguientes valores:

Nombre	Tipo	Descripción
<code>s_filesystem_type</code>	int	Guarda el número que identifica el sistema de archivos utilizado.
<code>s_inodes_count</code>	int	Guarda el número total de inodos
<code>s_blocks_count</code>	int	Guarda el número total de bloques
<code>s_free_blocks_count</code>	int	Contiene el número de bloques libres
<code>s_free_inodes_count</code>	int	Contiene el número de inodos libres
<code>s_mtime</code>	time	Última fecha en el que el sistema fue montado
<code>s_umtime</code>	time	Última fecha en que el sistema fue desmontado
<code>s_mnt_count</code>	int	Indica cuantas veces se ha montado el sistema

s_magic	int	Valor que identifica al sistema de archivos, tendrá el valor 0xEF53
s_inode_size	int	Tamaño del inodo
s_block_size	int	Tamaño del bloque
s_first_ino	int	Primer inodo libre
s_first_blo	int	Primer bloque libre
s_bm_inode_start	int	Guardará el inicio del bitmap de inodos
s_bm_block_start	int	Guardará el inicio del bitmap de bloques
s_inode_start	int	Guardará el inicio de la tabla de inodos
s_block_start	int	Guardará el inicio de la tabla de bloques

Esta información estará al inicio del sistema de archivos, no cambia de tamaño y se debe actualizar, según se vayan realizando las operaciones en el sistema de archivos. Por ejemplo al usar mount, debe actualizar s_mtime, al utilizar umount actualizará s_umtime, etc.

Journaling

Un sistema con journaling es un sistema de ficheros tolerante a fallos en el cual la integridad de los datos está asegurada porque las modificaciones de la meta-información de los ficheros son primero grabadas en un registro cronológico (log o journal, que simplemente es una lista de transacciones) antes que los bloques originales sean modificados. En el caso de un fallo del sistema, un sistema con journaling asegura que la consistencia del sistema de ficheros es recuperada. El método más común es el de grabar previamente cualquier modificación de la meta-información en un área especial del disco, el sistema realmente grabará los datos una vez que la actualización de los registros haya sido completada.

La journaling manejará todas las transacciones que realiza el sistema de Archivos EXT3 (respaldo para recuperación), la cual maneja la siguiente información.

Dato en la estructura	Descripción
Journal_tipo_operacion	El tipo de operación a realizarse
Journal_tipo	Si es archivo (0), si es directorio(1)
Journal_nombre	Nombre archivo o directorio

Journal_contenido	Si hay datos contenidos
Journal_fecha	Fecha de transacción
Journal_propietario	Es el propietario del archivo o directorio
Journal_Permisos	Son los permisos que tiene el archivo o directorio

Bitmap de Inodos

Indica que inodos están ocupados o libres, siendo los posibles valores 1 o 0.

1 = Inodo ocupado

0 = Inodo libre

Cada valor se guardará como un char.

Bitmap de Bloques

Indica que bloques están ocupados o libres, siendo los posibles valores 1 o 0.

1 = Bloque ocupado

0 = Bloque libre

Cada valor se guardará como un char.

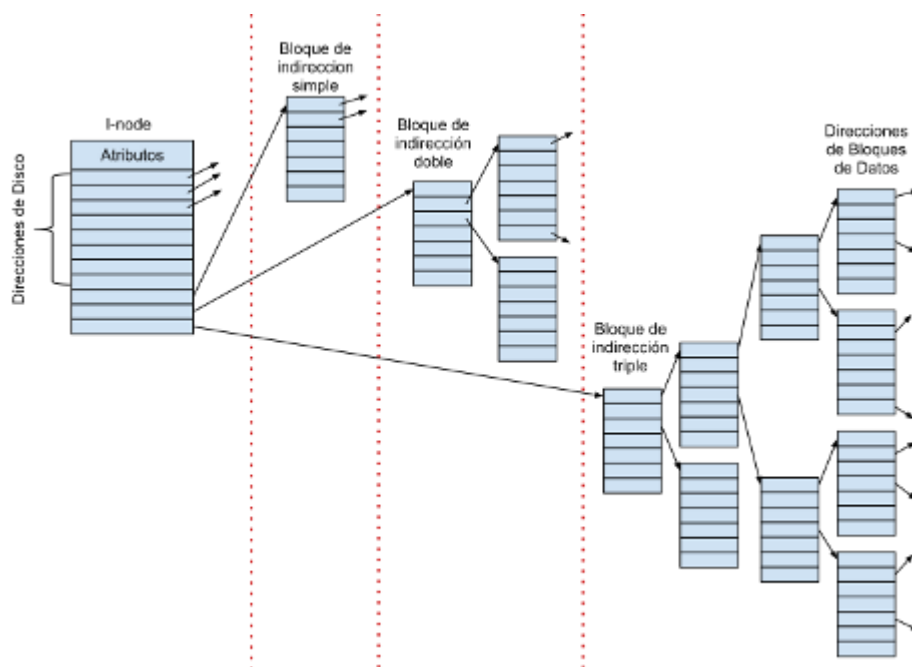
Tabla de inodos

Se utilizará un inodo por carpeta o archivo. Cada inodo tendrá la siguiente información:

Nombre	Tipo	Descripción
i_nlink	Int	El número de enlaces duros que tiene un archivo. Los enlaces simbólicos no se cuentan en el total.
i_pathlink	Char *	Enlaza directamente a la dirección del fichero.
i_uid	Int	UID del usuario propietario del archivo o carpeta
i_gid	Int	GID del grupo al que pertenece el archivo o carpeta.
i_size	Int	Tamaño del archivo en bytes
i_atime	Time	Última fecha en que se leyó el inodo sin modificarlo
i_ctime	Time	Fecha en la que se creó el inodo

i_mtime	Time	Última fecha en la que se modificó el inodo
i_block	int[15]	Array en los que los primeros 12 registros son bloques directos. El 13 será el número del bloque simple indirecto El 14 será el número del bloque doble indirecto El 15 será el número del bloque triple indirecto Si no son utilizados tendrá el valor -1
i_type	Char	Indica si es archivo, carpeta o enlace simbólico. Tendrá los siguientes valores: 0 = Archivo, 1 = Carpeta, 2= symlink.
i_perm	Int	Guardará los permisos del archivo o carpeta. Se trabajará a nivel de bits, estará dividido de la siguiente forma: Los primeros tres bits serán para el Usuario i_uid . Los siguientes tres bits serán para el Grupo al que pertenece el usuario. Y los últimos tres bits serán para los permisos de Otros usuarios . Cada grupo de tres bits significa lo siguiente: El primer bit indica el permiso de lectura R . El segundo bit indica el permiso de escritura W . El tercer bit indica el permiso de ejecución X .

La siguiente imagen muestra el funcionamiento de los bloques indirectos.



Bloque de Carpetas

Esta estructura estará asociada a un inodo de carpetas. Aquí se guardará la información sobre el nombre de los archivos que contiene y a que inodo apuntan. La estructura es la siguiente:

Nombre	Tipo	Descripción
b_content	content[4]	Array con el contenido de la carpeta

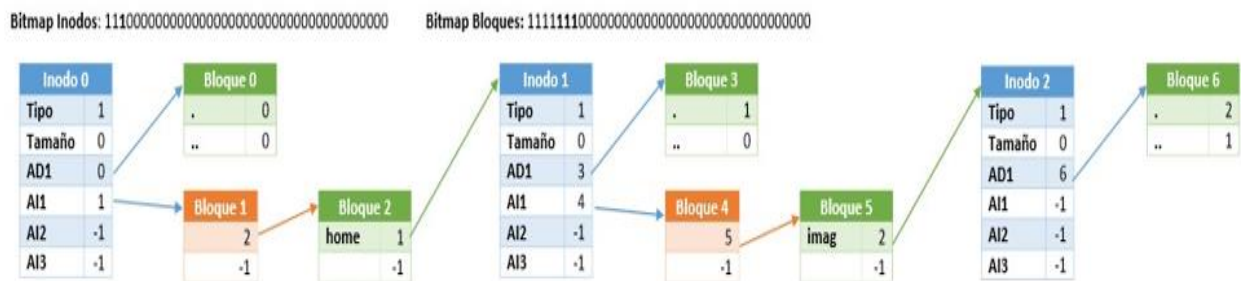
La estructura content será como la siguiente:

Nombre	Tipo	Descripción
b_name	char[12]	Nombre de la carpeta o archivo
b_inodo	int	Apuntador hacia un inodo asociado al archivo o carpeta

El tamaño de este bloque será de $4 * (12 + 4) = 64$ bytes. En cada inodo de carpeta, en el primer apuntador directo, en los primeros dos registros se guardará el nombre de la carpeta y su padre.

Ejemplos:

Por motivos de ejemplo se usaron bloques de carpeta (color verde) con capacidad de 2 ítems, bloques de apuntadores (color anaranjado) con capacidad de 2 apuntadores.



Bloque de Archivos

Este bloque guardará el contenido de un archivo. Su estructura es la siguiente:

Nombre	Tipo	Descripción
b_content	char[64]	Array con el contenido del archivo

Por lo que su tamaño, al igual que el bloque de carpetas, es de 64 bytes.

Bloque de Apuntadores

Estos bloques se utilizarán para los apuntadores indirectos (simples, dobles y triples). Su estructura es la siguiente:

Nombre	Tipo	Descripción
b_pointers	int[16]	Array con los apuntadores hacia bloques (de archivo o carpeta)

Su tamaño será de $16 * 4 = 64$, igual que los otros dos bloques anteriores.

El tipo de bloque que debe leer o utilizarse se puede determinar según el tipo de inodo (archivo o carpeta) y en base a qué apuntador esté utilizando (directo, simple, doble o triple indirecto)

Limitaciones

En esta sección se calcularán las limitantes del sistema descrito anteriormente. Primero se calculará el máximo de bloques que puede tener asociados un inodo.

$$\text{numero_bloques_por_inodo} = 12 + 16^1 + 16^2 + 16^3 = 12 + 16 + 256 + 4096 = 4380 \text{ bloques}$$

Cada bloque de carpeta tiene una capacidad de 4 hijos. Por lo que su capacidad es de 17520 carpetas o archivos dentro de una carpeta.

$$\text{capacidad_carpeta} = 4380 * 4 = 17520$$

Cada bloque de archivo tiene una capacidad de 64 bytes. Por lo que el tamaño máximo de un archivo es aproximadamente de 273 Kilobytes.

$$\text{capacidad_archivo} = 4380 * 64 = 280320 \text{ bytes} = 273 \text{ Kilobytes}$$

Al formatear se debe crear la carpeta raíz (/) y el archivo users.txt dentro de la raíz.

Otras operaciones

Aquí se aclararán algunas otras operaciones que se deben realizar. Por ejemplo que se utilizará el ajuste de la partición para buscar bloques libres y contiguos, correspondiente al momento de crear los archivos.

Al momento de modificar archivos pueden darse tres casos:

Ocupa más espacio: En este caso se utiliza el ajuste de la partición para buscar nuevos bloques contiguos y almacenar el contenido que exceda a los bloques ya utilizados. El contenido se escribe en los bloques que ya se están utilizando y el excedente en los bloques nuevos.

Ocupa menos espacio: Si utiliza menos bloques, únicamente los marcará como libres en el bitmap y eliminará las referencias hacia los bloques en los inodos o bloques de apuntadores indirectos.

Ocupa igual espacio: Si utiliza la misma cantidad, solo modifica los bloques.

En cualquiera de los casos anteriores debe modificarse el bitmap si es necesario y los datos del inodo (fecha de modificación, etc.)

Si un bloque de apuntadores indirectos queda vacío, se debe marcar como libre en el bitmap y quitar la referencia del inodo o bloque de apuntadores que lo estaba utilizando. Si un archivo ocupa **0 bytes** no tendrá bloque asociado.

Cada comando anterior debe modificar las características de los inodos según considere necesario, por ejemplo un cambio de permisos sobre el archivo modificará el campo **i_perm** del inodo.

El formateo fast, únicamente limpia con 0s los bitmap de inodos y bloques. El full aplica un formateo fast y además limpia los bloques e inodos. Siempre debe existir la carpeta raíz y el archivo de usuarios users.txt en la raíz.

Reportes

Se deberán generar los reportes con el comando rep. Este comando no necesita tener una sesión activa. Se generarán en graphviz. Se puede utilizar html dentro de los reportes si el estudiante lo considera necesario. Deberá mostrarlos de forma similar a los ejemplos mostrados y deben poder visualizarse tan pronto como se ejecute el comando y no ir a buscar el reporte, ya que tiene que ser automatizado este proceso.

IMPORTANTE: Esta parte es **obligatoria** para tener derecho a la calificación de los aspectos que muestre el reporte. Si falta alguno de los reportes no se calificará. Por ejemplo si no hace reporte de inodos, no tendrá derecho a la calificación de todos los aspectos relativos a los inodos, ya que no se puede comprobar que el estudiante haya implementado dicha funcionalidad.

rep

Recibirá el nombre del reporte que se desea y lo generará con graphviz en una carpeta existente.

Parámetro	Categoría	Descripción
-name	Obligatorio	Nombre del reporte a generar. Tendrá los siguientes valores: <ul style="list-style-type: none">• mbr• disk• inode• Journaling• block• bm_inode• bm_block• tree• sb• file• Ls +i• Ls +l Si recibe otro valor que no sea alguno de los anteriores, debe mostrar un error.
-path	Obligatorio	Indica una carpeta y el nombre que tendrá el reporte. Si no existe la carpeta, deberá crearla.
-id	Obligatorio	Indica el id de la partición que se utilizará. Si el reporte es sobre la información del disco, se utilizará el disco al que pertenece la partición. Si no existe debe mostrar un error.
+ruta	Opcional	Funcionará para el reporte file y ls +i / +l. Será el nombre del archivo o carpeta del que se mostrará el reporte. Si no existe muestra error.

Ejemplos

```
rep -id::vda1 -Path::"/home/user/reports/reporte1.jpg" \ -name::mbr
rep -id::vda2 -Path::"/home/user/reports/reporte2.png" \ -name::disk
rep -id::vda2 -Path::"/home/user/reports/reporte 2.png" \
    -name::ls+i +ruta::"/home/mis documentos"
rep -id::vda1 -Path::"/home/user/reports/reporte 3.jpg" -name::tree
```

mbr

Mostrará tablas con **toda** la información del MBR, así como de los EBR que se han creado.

Ejemplo:

MBR Disco1.dsk

Nombre	Valor
mbr_tamaño	10485760
mbr_fecha_creacion	30/07/2016 13:45
mbr_disk_signature	16684811
part_status_1	1
part_type_1	P
part_fit_1	B
part_start_1	300
part_size_1	1048576
part_name_1	Partición 1
part_status_2	1
part_type_2	E
part_fit_2	W
part_start_2	1048876
part_size_2	1048576
part_name_2	Particion 2

EBR_1

Nombre	Valor
part_status_1	1
part_fit_1	F
part_start_1	1048876
part_size_1	524438
part_next_1	1573314
part_name_1	Logica 1

EBR_2

Nombre	Valor
part_status_1	1
part_fit_1	B
part_start_1	1573514
part_size_1	524438
part_next_1	-1
part_name_1	Logica 2

disk

Este reporte mostrará la estructura de las particiones (mostrar nombre de la partición, el tipo, el sistema de archivos....) y el mbr del disco.

Ejemplo:

Disco1.dsk

MBR	Libre	Extendida					Primaria	Libre
		EBR	Lógica	Libre	EBR	Lógica		

Reporte de Journaling

Se debe escribir en un archivo de texto toda la información que está almacenada en la bitácora, el formato **SUGERIDO**:

Tipo Operación: "El tipo de operación a realizarse"

Tipo: "Archivo/Directorio"

Nombre: "Nombre archivo o directorio"

Contenido: "Si hay datos contenidos"

Fecha "Fecha de transacción"

Propietario: "Usuario"

Permisos: 777

Tipo Operación: "El tipo de operación a realizarse"

Tipo: "Archivo/Directorio"

Nombre: "Nombre archivo o directorio"

Contenido: "Si hay datos contenidos"

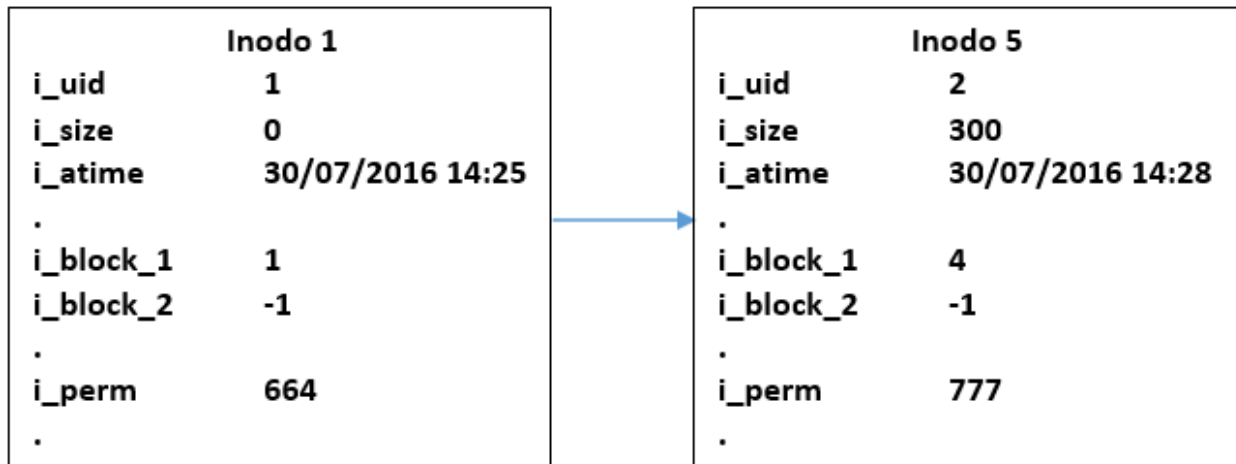
Fecha "Fecha de transacción"

Propietario: "Usuario"

Permisos: 777

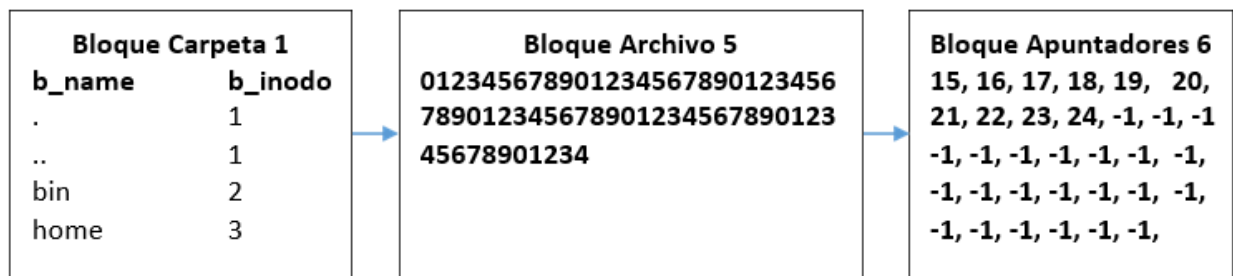
inode

Mostrará bloques con **toda** la información de los inodos **utilizados**. Si no están utilizados no debe mostrarlos.



Block

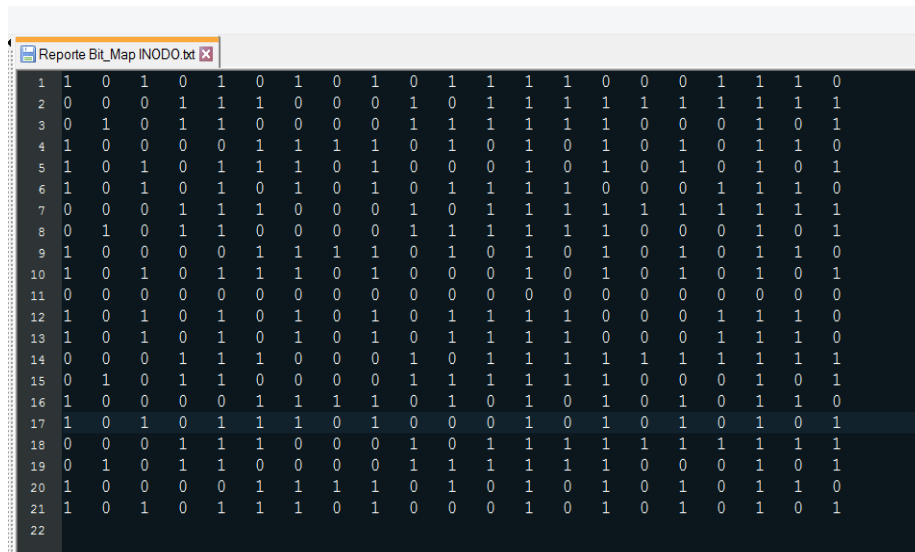
Mostrará la información de todos los bloques **utilizados**. Si no están utilizados no debe mostrarlos.



bm_inode

Este reporte mostrará la información del bit map de inodos, mostrará todos los bits, libres o utilizados, este reporte se generará en un archivo de texto mostrando 20 registros por línea.

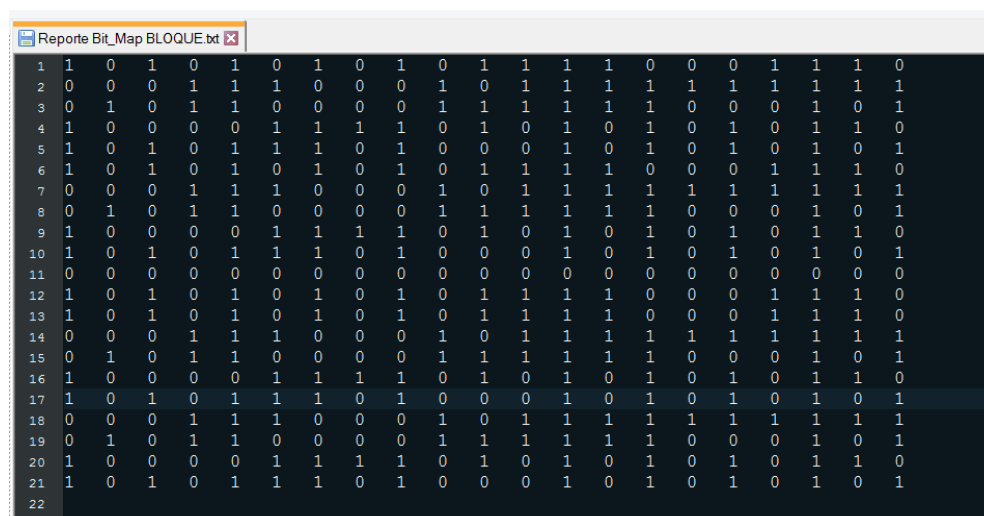
Ejemplo:



1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	0	0	1	1	1	0
2	0	0	0	1	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1
3	0	1	0	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	1	0	1
4	1	0	0	0	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	0
5	1	0	1	0	1	1	1	0	1	0	0	0	1	0	1	0	1	0	1	0	1
6	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	0	0	1	1	1	0
7	0	0	0	1	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1
8	0	1	0	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	1	0	1
9	1	0	0	0	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	0
10	1	0	1	0	1	1	1	0	1	0	0	0	1	0	1	0	1	0	1	0	1
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	0	0	1	1	1	0
13	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	0	0	1	1	1	0
14	0	0	0	1	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1
15	0	1	0	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	1	0	1
16	1	0	0	0	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	0
17	1	0	1	0	1	1	1	0	1	0	0	0	1	0	1	0	1	0	1	0	1
18	0	0	0	1	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1
19	0	1	0	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	1	0	1
20	1	0	0	0	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	0
21	1	0	1	0	1	1	1	0	1	0	0	0	1	0	1	0	1	0	1	0	1
22																					

bm_block

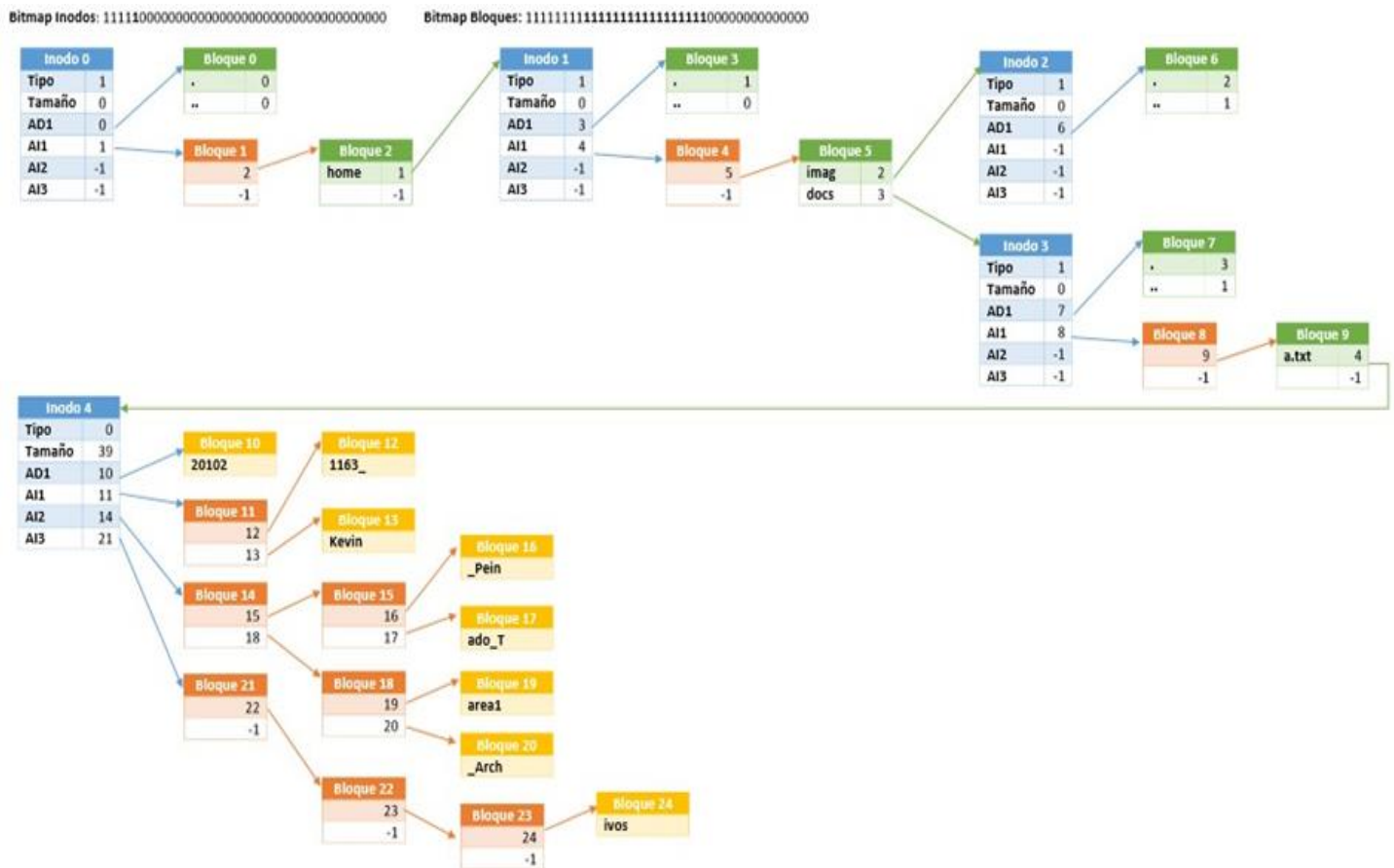
Este reporte mostrará la información del bitmap de bloques, mostrará todos los bits, libres o utilizados, este reporte se generará en un archivo de texto que mostrará 20 registros por línea.



1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	0	0	1	1	1	0
2	0	0	0	1	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1
3	0	1	0	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	1	0	1
4	1	0	0	0	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	0
5	1	0	1	0	1	1	1	0	1	0	0	0	1	0	1	0	1	0	1	0	1
6	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	0	0	1	1	1	0
7	0	0	0	1	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1
8	0	1	0	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	1	0	1
9	1	0	0	0	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	0
10	1	0	1	0	1	1	1	0	1	0	0	0	1	0	1	0	1	0	1	0	1
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	0	0	1	1	1	0
13	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	0	0	1	1	1	0
14	0	0	0	1	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1
15	0	1	0	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	1	0	1
16	1	0	0	0	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	0
17	1	0	1	0	1	1	1	0	1	0	0	0	1	0	1	0	1	0	1	0	1
18	0	0	0	1	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1
19	0	1	0	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	1	0	1
20	1	0	0	0	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	0
21	1	0	1	0	1	1	1	0	1	0	0	0	1	0	1	0	1	0	1	0	1
22																					

tree

Este reporte genera el arbol de **todo** el sistema ext2/ext3. Se mostrará **toda** la información de los inodos o bloques. No deben ponerse los bloques o inodos libres, únicamente se pondrán los bloques que están siendo utilizados. Deberá ser como el siguiente (En este ejemplo no se ponen todos los datos, bloques y flechas por falta de espacio, se utilizaron bloques de carpeta con capacidad 2, bloques de apuntadores con capacidad 2 y bloques de archivo con capacidad 5):



sb

Muestra **toda** la información del superbloque en una tabla.

Ejemplo:

SuperBloque Partición 1 en Disco1.dsk

Nombre	Valor
s_inodes_count	200
s_blocks_count	600
s_free_blocks_count	10
s_free_inodes_count	100
s_mtime	30/07/2016 15:38
s_umtime	30/07/2016 15:38
s_mnt_count	4
s_magic	0xEF53
s_inode_size	128
s_block_size	64
s_first_ino	50
s_first_blo	180
s_bm_inode_start	128
s_bm_block_start	328
s_inode_start	630
s_block_start	15852

file

Este reporte muestra el nombre y **todo** el contenido del archivo especificado en el parámetro **file**.

Ejemplo:

a.txt

```
012345678901234567890123456789012345678901234567890123456789012345678901
234567890123456789012345678901234567890123456789012345678901234567890123
456
```

Ls+l

Este reporte mostrará la información de los archivos y carpetas con permisos, fecha de modificación, tipo, fecha de creación y número de enlaces.

Permisos	owner	Links	grupo	Size	Fecha	Hora	Name
-rw-rw-r--	user1	1	Mi grupo	40661	24/02/2016	09:53	Manual.txt
-rw-r--rwx	user2	2	Mi grupo	123	24/02/2016	09:53	archivo1.txt
drw-rw-r--	root	3	root	0	24/02/2016	09:53	home

Ls+i

Este reporte mostrará la información de los archivos y carpetas con permisos, fecha de modificación, tipo, fecha de creación, el número de inodo y número de enlaces.

Inodo	Permisos	owner	Links	grupo	Size	Fecha	Hora	Name
98921	-rw-rw-r--	user1	1	grupo	4066	24/02/2016	09:53	Manual.txt
23423	lrw-r--rwx	user2	2	grupo	123	24/02/2016	09:53	enlace-o1.txt -> o1.txt
12434	drw-rw-r--	root	3	root	0	24/02/2016	08:13	home

Scripts

Son archivos con los comandos definidos en este documento. También puede haber comentarios y líneas en blanco.

Ejemplo

Calificación.sh

#Contenido de calificacion.sh

#Crea tres discos de 30 Mb

```
mkdisk -Size::30 +unit::M -path::"/home/Disco1.dsk" mdkisk +unit::K -path::"/home/Disco2.dsk" -size::30720
```

```
mkDisk -size::30 -patH::"/home/archivos/Disco3.dsk"
```

#Debería dar error

```
mkDisk -param::x -size::30 -patH::"/home/Disco4.dsk"
```

#Elimina un disco, el primero debería dar error rmDisk -

```
patH::"/home/Disco4.dsk"
```

```
rmDisk -patH::"/home/archivos/Disco3.dsk"
```

#Crear particiones

```
fdisk -Size::10240 -path::"/home/Disco1.dsk" -name::"Part1"
```

```
fdisk +type::E -path::"/home/Disco1.dsk" +Unit::K \
```

```
-name::Part2 -size::10240
```

```
fdisk +type::L -path::"/home/Disco1.dsk" +Unit::K \
```

```
-name::Part3 -size::10240
```

```
fdisk -Size::10240 -path::"/home/Disco2.dsk" -name::"Part1"
```

```
fdisk -Size::10240 -path::"/home/Disco2.dsk" -name::"Part1"
```

#Formatear particiones

```
mkfs -name::Part1 -path::"/home/Disco1.dsk"
```

```
mkfs -path::"/home/Disco1.dsk" -name::Part2
```

```
mkfs -path::"/home/Disco2.dsk" -name::Part1 +fit::BF
```

```
mkfs -path::"/home/Disco2.dsk" -name::Part2
```

#Agrega 500 kb a Part1

```
mkfs -add::500 path::"/home/Disco3.dsk" -name::Particion3
```

#Monta las particiones

```
mount -path::"/home/Disco1.dsk" -name::Part1 #id::vda1
```

```
mount -path::"/home/Disco2.dsk" -name::Part1 #id::vdb1
```

```
mount -path::"/home/Disco1.dsk" -name::Part2 #id::vda2
```

#loguearse

```
login -usr::root -pwd::123
```

#Crear grupos y usuarios

```
mkgrp -id::vda1 -name::usuarios
```

```
Mkusr -id::vda1 -name::user1 -grp::usuarios -pwd::12345
```

```
Mkusr -id::vda1 -name::user2 -grp::usuarios -pwd::12345
```

```
#salir loguearse
logout
#Error
login -usr::user1 -pwd::123
login -usr::user1 -pwd::12345
```

```
#Crear carpetas
mkdir -p -id::vda1 -path::"/home/user1/documents/a"
mkdir -p -id::vdb2 -path::"/home/user1/documents/b"
```

```
#Error, ya existe
mkdir -id::vda1 -path::"/home/user1/documents/a"
mkdir -id::vda1 -path::"/home/user1/documents/b"
mkdir -id::vda1 -path::"/home/user1/documents/c"
mkdir -id::vda1 -path::"/home/user1/documents/d"
mkdir -id::vda1 -path::"/home/user1/documents/e"
mkdir -id::vda1 -path::"/home/user1/documents/f"
mkdir -id::vda1 -path::"/home/user1/documents/g"
mkdir -id::vda1 -path::"/home/user1/documents/h"
mkdir -id::vda1 -path::"/home/user1/documents/i"
mkdir -id::vda1 -path::"/home/user1/documents/j"
mkdir -id::vda1 -path::"/home/user1/documents/k"
mkdir -id::vda1 -path::"/home/user1/documents/l"
mkdir -id::vda1 -path::"/home/user1/documents/m"
mkdir -id::vda1 -path::"/home/user1/documents/n"
mkdir -id::vda1 -path::"/home/user1/documents/o"
mkdir -id::vda1 -path::"/home/user1/documents/p"
mkdir -id::vda1 -path::"/home/user1/documents/q"
```

```
#Crear archivos
mkfile -size::51200 -id::vda1 -path::"/home/user1/documents/a.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/b.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/c.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/d.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/e.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/f.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/g.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/h.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/i.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/j.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/k.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/l.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/m.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/n.txt"
mkfile -size::1024 -id::vda1 -path::"/home/user1/documents/o.txt"
```

.....

.....

exec

El programa podrá ejecutar scripts con el comando exec. Debe mostrar el contenido de la línea que está leyendo y su resultado. También debe mostrar los comentarios del script.

Parámetro	Categoría	Descripción
-path	Obligatorio	Especifica el nombre del script que se va a ejecutar.

Ejemplo:

#ejecuta el script

exec /home/Desktop/calificacion.sh

Entrega

FASE 1

La fase 1 se entregará el día viernes 12 de agosto antes de las **23:50 PM**, al correo deben enviar el link de su repositorio en github, para su descarga (tomen en cuenta que se ve la fecha de la última modificación). Asunto: [MIA]Fase1_carné

Pares: 201212838@ingenieria.usac.edu.gt e **Impares:** Georgina.estrada78@gmail.com

- | | |
|----------------------------|--|
| 1. Administración de disco | 2. Reportes (Esenciales para calificación) |
| 1.1. Mkdisk | 2.1. Mbr |
| 1.2. Rmdisk | 2.2. Disk |
| 1.3. Fdisk | 2.3. Exec |
| 1.4. Mount | |
| 1.5. Umount | |

FASE 2

Todas las operaciones no marcadas en la fase 1 deben entregarse. Como requisito debe haber entregado la Fase 1, de lo contrario no tendrá derecho a entregar la fase 2. La fecha de entrega es viernes 16 de septiembre antes de las **23:50 PM**, al correo deben enviar el link de su repositorio en github, para su descarga (tomen en cuenta que se ve la fecha de la última modificación). Asunto: [MIA] Fase2_carné.

Para ambos casos se deberán tener en el repositorio su código fuente y **ejecutable funcional (es indispensable ya que con este se realizará la calificación)**.

- La fase 3 se enviará el día viernes 7 de octubre.
- El lenguaje a utilizar es C y el IDE codeblocks, de no usar estos no se calificará.
- Solo se calificará sobre una instalación física de una distribución GNU/Linux. De no ser así se penalizará la nota obtenida.
- No se permite la modificación de código durante la calificación. El estudiante no tendrá acceso al código fuente durante la calificación.
- El archivo binario que representa a los discos no debe crecer.
- No se permite la utilización de estructuras en memoria (listas, arboles, etc.) para el manejo de los archivos o carpetas.
- **Para calificarse como Mínimo deberán tener**
 - **Fase1:** comandos, reportes, exec, creación de particiones (primarias)
 - **Fase 2:** toda la fase 1, permisos, mkfs, mkfile, mkdir, reportes, convert ext2 to ext3, perdida y recuperación del sistema ext3.