

1er cuatrimestre de 2019

Sistemas Operativos y  
Redes II

**Trabajo Práctico N° 1**  
**FAT**

Docente:  
Alexis Tcach

Integrantes:

Nicolás Di  
Biase

Respuestas:

- 1) Se ha puesto umask=000 para permitir que todos los usuarios tengan permisos de escritura, lectura y ejecución, en este caso, sobre la imagen montada.
- 2) a) Hay 4 particiones, de las cuales la primera corresponde a una partición de tipo FAT12 y las otras tres a particiones vacías.

A continuación se pueden ver los primeros bytes del MBR mediante el Hexeditor:

```
00000000 EB 3C 90 6D 6B 66 73 2E 66 61 74 00 02 04 01 00 .<.mkfs.fat.....
00000010 02 00 02 00 08 F8 02 00 20 00 40 00 00 00 00 00 ..... @.....
00000020 00 00 00 00 80 01 29 5F 05 C8 06 4E 4F 20 4E 41 .....)_...NO NA
00000030 4D 45 20 20 20 20 46 41 54 31 32 20 20 20 0E 1F ME FAT12 ..
00000040 BE 5B 7C AC 22 C0 74 0B 56 B4 0E BB 07 00 CD 10 .[|."t.V.....
00000050 5E EB F0 32 E4 CD 16 CD 19 EB FE 54 68 69 73 20 ^..2.....This
00000060 69 73 20 6E 6F 74 20 61 20 62 6F 6F 74 61 62 6C is not a bootabl
00000070 65 20 64 69 73 6B 2E 20 20 50 6C 65 61 73 65 20 e disk. Please
00000080 69 6E 73 65 72 74 20 61 20 62 6F 6F 74 61 62 6C insert a bootabl
00000090 65 20 66 6C 6F 70 70 79 20 61 6E 64 0D 0A 70 72 e floppy and..pr
000000A0 65 73 73 20 61 6E 79 20 6B 65 79 20 74 6F 20 74 ess any key to t
000000B0 72 79 20 61 67 61 69 6E 20 2E 2E 2E 20 0D 0A 00 ry again ... ..
```

En la siguiente imagen se pueden ver las 4 particiones:

```
00 00 00 00 00 00 00 00 00 00 00 00 00 00 80 00
02 00 01 20 20 00 01 00 00 00 FF 07 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 AA
```

En verde está marcada la primer partición, la cual es de tipo FAT12, y con amarillo las otras tres particiones vacías.

- b) El archivo “read\_boot.c” muestra los datos del Boot Sector

```

Partition type: 1
Encontrado FAT12 en posición 0
  Jump code: EB:3C:90
  OEM code: [mkfs.fat]
  sector_size: 512
  sectors_per_cluster: 4
  reserved_sectors: 1
  number_of_fat: 2
  root_dir_entries: 512
  number_of_sectors: 2048
  media_type: 0xF8
  fat_size_sectors: 2
  sectors_per_track: 32
  number_of_head: 64
  hidden_sectors: 0
  total_sectors_long: 0
  drive_number: 0x80
  reserved: 0x01
  boot_signature: 0x29
  volume_id: 0x4E0F6457
  Volume label: [NO NAME    ]
  Filesystem type: [FAT12   ]
  Boot sector signature: 0xAA55

```

c) La primera partición es booteable, el primer byte con valor 0x80 indica que es booteable.

d) El archivo “read\_mbr.c” muestra los datos de las cuatro particiones, si es booteable, comienzo de partición en CHS, el tipo de partición, fin de partición en CHS y tamaño en sectores:

```

Partition entry 0: First byte 80
  Comienzo de partición en CHS: 00:02:00
  Partition type 0x01
  Fin de partición en CHS: 00:20:20
  Dirección LBA relativa 0x00000001, de tamaño en sectores 2047
Partition entry 1: First byte 00
  Comienzo de partición en CHS: 00:00:00
  Partition type 0x00
  Fin de partición en CHS: 00:00:00
  Dirección LBA relativa 0x00000000, de tamaño en sectores 0
Partition entry 2: First byte 00
  Comienzo de partición en CHS: 00:00:00
  Partition type 0x00
  Fin de partición en CHS: 00:00:00
  Dirección LBA relativa 0x00000000, de tamaño en sectores 0
Partition entry 3: First byte 00
  Comienzo de partición en CHS: 00:00:00
  Partition type 0x00
  Fin de partición en CHS: 00:00:00
  Dirección LBA relativa 0x00000000, de tamaño en sectores 0

```

3) a) Mediante el Hexeditor podemos ver que existen los siguientes archivos/entradas:

Los que se encuentran en el root directory:

```
00000A00  41 6D 00 69 00 5F 00 64 00 69 00 0F 00 D0 72 00 Am.t._.d.t....r.
00000A10  00 00 FF FF FF FF FF FF FF FF 00 00 FF FF FF FF .....
00000A20  4D 49 5F 44 49 52 20 20 20 20 20 10 00 00 97 BA MI_DIR .....
00000A30  7B 4A 7B 4A 00 00 97 BA 7B 4A 03 00 00 00 00 00 {J{J....{J.....
00000A40  41 68 00 6F 00 6C 00 61 00 2E 00 0F 00 A0 74 00 Ah.o.l.a.....t.
00000A50  78 00 74 00 00 00 FF FF FF FF 00 00 FF FF FF FF x.t.....
00000A60  48 4F 4C 41 20 20 20 20 54 58 54 20 00 64 F3 7C HOLA TXT .d.|
00000A70  7A 4A 95 4E 00 00 F3 7C 7A 4A 05 00 3C 00 00 00 zJ.N...|zJ.<...
00000A80  41 70 00 72 00 75 00 65 00 62 00 0F 00 83 61 00 Ap.r.u.e.b....a.
00000A90  2E 00 74 00 78 00 74 00 00 00 00 00 FF FF FF FF ..t.x.t.....
00000AA0  50 52 55 45 42 41 20 20 54 58 54 20 00 00 90 29 PRUEBA TXT ...
00000AB0  95 4E 95 4E 00 00 90 29 95 4E 04 00 07 00 00 00 .N.N...)N.....
00000AC0  E5 4F 52 52 41 52 7E 31 53 57 50 20 00 00 F3 B4 .ORRAR~1SWP ....
00000AD0  7B 4A 7B 4A 00 00 F3 B4 7B 4A 00 00 00 00 00 00 {J{J....{J.....
00000AE0  E5 2E 00 73 00 77 00 78 00 00 00 0F 00 A9 FF FF ...s.w.x.....
00000AF0  FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF .....
00000B00  E5 2E 00 62 00 6F 00 72 00 72 00 0F 00 A9 61 00 ...b.o.r.r....a.
00000B10  72 00 6F 00 6E 00 2E 00 74 00 00 00 78 00 74 00 r.o.n...t...x.t.
00000B20  E5 4F 52 52 41 52 7E 31 53 57 58 20 00 00 F3 B4 .ORRAR~1SWX ....
00000B30  7B 4A 7B 4A 00 00 F3 B4 7B 4A 00 00 00 00 00 00 {J{J....{J.....
```

Los que se encuentran dentro del subdirectorio MI\_DIR:

```
00005200  2E 20 20 20 20 20 20 20 20 20 20 10 00 00 91 BA . .....
00005210  7B 4A 7B 4A 00 00 91 BA 7B 4A 03 00 00 00 00 00 {J{J....{J.....
00005220  2E 2E 20 20 20 20 20 20 20 20 20 10 00 00 91 BA .. .....
00005230  7B 4A 7B 4A 00 00 91 BA 7B 4A 00 00 00 00 00 00 {J{J....{J.....
00005240  41 76 00 61 00 63 00 69 00 6F 00 0F 00 5B 2E 00 Av.a.c.i.o...[..
00005250  74 00 78 00 74 00 00 00 FF FF 00 00 FF FF FF FF t.x.t.....
00005260  56 41 43 49 4F 20 20 20 54 58 54 20 00 00 97 BA VACIO TXT ....
00005270  7B 4A 95 4E 00 00 97 BA 7B 4A 00 00 00 00 00 00 {J.N....{J.....
```

Y mediante el código generado podemos ver los siguientes archivos:

```
Root dir_entries 512
Archivo: [Am.]
Directorio: [. ]
Directorio: [.. ]
Archivo: [Av.]
Archivo: [VACIO .TXT]
Directorio: [MI_DIR . ]
Archivo: [Ah.]
Archivo: [HOLA .TXT]
Archivo borrado: [?p.]
Archivo borrado: [?RUEBA .TXT]
Archivo borrado: [?ORRAR~.SWP]
Archivo borrado: [?..]
Archivo borrado: [?..]
Archivo borrado: [?ORRAR~.SWX]
```

El código que se programó, consiste en leer todas las entradas del root directory y si una entrada corresponde a un directorio se lee en primera instancia las entradas dentro del mismo y se las va mostrando por pantalla, luego se muestra el directorio y el resto de las entradas. Para mostrar las entradas de un subdirectorio se hace un seek a la posición correspondiente al cluster donde están las entradas del subdirectorio leído.

b) A continuación se muestra el archivo “prueba.txt”.

```
root@nicodibi-VirtualBox:/mnt# ls
hola.txt  mi_dir  prueba.txt
```

```
00000A00  41 6D 00 69 00 5F 00 64 00 69 00 0F 00 D0 72 00 Am.i._.d.i....r.
00000A10  00 00 FF FF FF FF FF FF FF FF 00 00 FF FF FF FF .....
00000A20  4D 49 5F 44 49 52 20 20 20 20 20 10 00 00 97 BA MI_DIR .....
00000A30  7B 4A 7B 4A 00 00 97 BA 7B 4A 03 00 00 00 00 00 {J{J....{J.....
00000A40  41 68 00 6F 00 6C 00 61 00 2E 00 0F 00 A0 74 00 Ah.o.l.a.....t.
00000A50  78 00 74 00 00 00 FF FF FF FF 00 00 FF FF FF FF x.t.....
00000A60  48 4F 4C 41 20 20 20 20 54 58 54 20 00 64 F3 7C HOLA   TXT .d.|
00000A70  7A 4A 95 4E 00 00 F3 7C 7A 4A 05 00 3C 00 00 00 zJ.N...|zJ...<...
00000A80  41 70 00 72 00 75 00 65 00 62 00 0F 00 83 61 00 Ap.r.u.e.b....a.
00000A90  2E 00 74 00 78 00 74 00 00 00 00 00 FF FF FF FF ..t.x.t.....
00000AA0  50 52 55 45 42 41 20 20 54 58 54 20 00 00 90 29 PRUEBA TXT ...)
00000AB0  95 4E 95 4E 00 00 90 29 95 4E 04 00 07 00 00 00 .N.N...).N.....
00000AC0  E5 4F 52 52 41 52 7E 31 53 57 50 20 00 00 F3 B4 .ORRAR~1SWP ....
00000AD0  7B 4A 7B 4A 00 00 F3 B4 7B 4A 00 00 00 00 00 00 {J{J....{J.....
00000AE0  E5 2E 00 73 00 77 00 78 00 00 00 0F 00 A9 FF FF ...s.w.x.....
00000AF0  FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF .....
00000B00  E5 2E 00 62 00 6F 00 72 00 72 00 0F 00 A9 61 00 ...b.o.r.r....a.
00000B10  72 00 6F 00 6E 00 2E 00 74 00 00 00 78 00 74 00 r.o.n...t...x.t.
00000B20  E5 4F 52 52 41 52 7E 31 53 57 58 20 00 00 F3 B4 .ORRAR~1SWX ....
00000B30  7B 4A 7B 4A 00 00 F3 B4 7B 4A 00 00 00 00 00 00 {J{J....{J.....
```

Luego el mismo se borró y desde el Hexeditor se puede ver de la siguiente manera:

```
00000AA0  E5 52 55 45 42 41 20 20 54 58 54 20 00 00 90 29 .RUEBA TXT ...)
00000AB0  95 4E 95 4E 00 00 90 29 95 4E 04 00 07 00 00 00 .N.N...).N.....
```

Y con el código generado anteriormente se ve así:

```
Root dir_entries 512
Archivo: [Am.]
Directorio: [.      .  ]
Directorio: [..     .  ]
Archivo: [Av.]
Archivo: [VACIO   .TXT]
Directorio: [MI_DIR .  ]
Archivo: [Ah.]
Archivo: [HOLA    .TXT]
Archivo borrado: [?♦p.]
Archivo borrado: [?♦RUEBA .TXT]
Archivo borrado: [?♦ORRAR~.SWP]
Archivo borrado: [?♦..]
Archivo borrado: [?♦..]
Archivo borrado: [?♦ORRAR~.SWX]
```

c) Para ver el archivo, a partir de la documentación leída, se sabe que un archivo o entrada borrada tiene como primer carácter de su nombre de archivo el equivalente al Hexadecimal 0xE5.

A continuación se puede ver, mediante Hexeditor, la entrada eliminada:

```
00000AA0  E5 52 55 45 42 41 20 20 54 58 54 20 00 00 90 29 .RUEBA TXT ...)
00000AB0  95 4E 95 4E 00 00 90 29 95 4E 04 00 07 00 00 00 .N.N...).N.....
```

Mediante el código generado anteriormente se puede ver el archivo eliminado:

```
Root dir_entries 512
Archivo: [Am.]
Directorio: [.      .  ]
Directorio: [..     .  ]
Archivo: [Av.]
Archivo: [VACIO  .TXT]
Directorio: [MI_DIR .  ]
Archivo: [Ah.]
Archivo: [HOLA   .TXT]
Archivo borrado: [?p.]
Archivo borrado: [?RUEBA .TXT]
Archivo borrado: [?ORRAR~.SWP]
Archivo borrado: [?..]
Archivo borrado: [?..]
Archivo borrado: [?ORRAR~.SWX]
```

d) Acerca del recupero de archivos, se puede decir que puede recuperarse siempre y cuando el tamaño del archivo no supere el tamaño de un cluster, ya que dentro de la estructura de entrada de FAT12 siempre tenemos en que cluster comienza el contenido del archivo, pero el resto del contenido se encuentra en la tabla FAT, la cual cuando se elimina el archivo también se eliminan los punteros a los clusters que tienen su contenido. Además se puede decir que el archivo se va a poder recuperar siempre y cuando no se inserte una nueva entrada que la pise, ya que para el File System es una entrada “vacía”.

- 4) Se creo el archivo “lapapa.txt”:

```
root@nicodibi-VirtualBox:/mnt# ls
hola.txt  lapapa.txt  mi_dir
```

Con el siguiente contenido:

```
root@nicodibi-VirtualBox:/mnt# more lapapa.txt
Archivo creado para el punto 4) del TP1 de SOR II
```

Mediante Hexeditor se visualiza así:

```
00000AA0  4C 41 50 41 50 41 20 20 54 58 54 20 00 00 70 BD  LAPAPA  TXT  ..p.
00000AB0  98 4E 98 4E 00 00 70 BD 98 4E 04 00 32 00 00 00  .N.N..p..N..2...
```

Y con el código generado previamente generado:



```

Root dir_entries 512
Archivo: [Am.]
Directorio: [ . ]
Directorio: [.. ]
Archivo: [Av.]
Archivo: [VACIO .TXT]
Directorio: [MI_DIR . ]
Archivo: [Ah.]
Archivo: [HOLA .TXT]
Archivo: [Al.]
Archivo: [LAPAPA .TXT]
Archivo borrado: [?ϕORRAR~.SWP]
Archivo borrado: [?ϕ..]
Archivo borrado: [?ϕ..]
Archivo borrado: [?ϕORRAR~.SWX]

```

b) El contenido en el Hexeditor se visualiza de la siguiente manera:

```

00005A00  41 72 63 68 69 76 6F 20 63 72 65 61 64 6F 20 70  Archivo creado p
00005A10  61 72 61 20 65 6C 20 70 75 6E 74 6F 20 34 29 20  ara el punto 4)
00005A20  64 65 6C 20 54 50 31 20 64 65 20 53 4F 52 20 49  del TP1 de SOR I
00005A30  49 0A 00 00 00 00 00 00 00 00 00 00 00 00 00 00  I.....

```

Se creó código en C (**mostrar\_contenido.c**), en la que por cada entrada del root directory se pregunta si el archivo es “lapapa.txt”. Para esto, se pregunta si el filename es LAPAPA y la extensión es TXT. Se compara con caracteres en mayúscula, ya que el file system los almacena en mayúscula. Luego se lee cada byte hasta llegar al tamaño del archivo.

Al ejecutar el código se visualiza de la siguiente manera:

```

Leido archivo lapapa.txt, en 0x5A00
Contenido del archivo: Archivo creado para el punto 4) del TP1 de SOR II

```

c) Se creó el archivo “**buscar\_recuperar\_archivo.c**”, que tiene la lógica para buscar una subcadena de una cadena. Si encuentra la subcadena pregunta si el archivo se encuentra eliminado (Se verifica esto mediante el contenido del primer carácter del filename) y en caso de estarlo se escribe nuevamente el archivo cambiando el primer carácter por una ‘A’. Para escribir sobre la misma ubicación de disco se almacenan el puntero del inicio del archivo para luego volver a dicha posición.