

## Nivel 6: `ficheros_basico.c` {`liberar_inodo()`, `liberar_bloques_inodo()`}, `ficheros.c` {`mi_truncar_f()`}, `truncar.c`

En el nivel 3 ya realizamos las funciones `leer_inodo()`, `escribir_inodo()` y `reservar_inodo()`, ahora vamos a realizar la que nos faltaba para completar el grupo: `liberar_inodo()`, la cual requiere de una función auxiliar para liberar **todos los bloques lógicos** que cuelgan del inodo: `liberar_bloques_inodo()`. Con ello completaremos el conjunto de funciones de la capa `ficheros_basico.c`.

Después realizaremos la función `mi_truncar_f()`, de la capa de ficheros, que también utiliza la función `liberar_bloques_inodo()` pero para liberar bloques **a partir de cualquier bloque lógico**.

Finalmente implementaremos un programa externo ficticio, `truncar.c`<sup>1</sup>, para poder llamar provisionalmente desde la consola a `liberar_inodo()` o a `mi_truncar_f()` dependiendo de si se pretende eliminar el inodo completo (en tal caso se liberarían todos los bloques del inodo a partir del bloque lógico 0) o si se pretende truncarlo a partir de cierto byte (en tal caso se liberarían solo los bloques restantes).

### 1) `int liberar_inodo(unsigned int ninodo);`

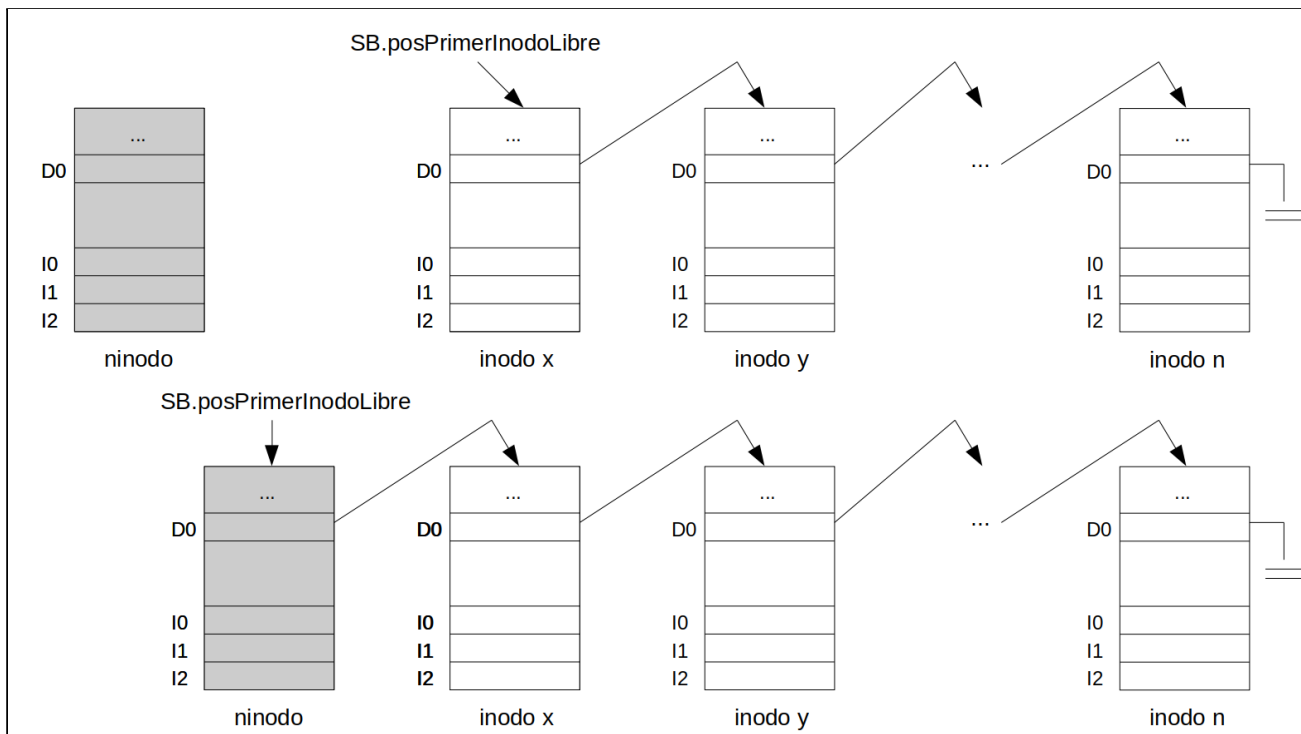
Liberar un inodo implica por un lado, que tal inodo pasará a la cabeza de la lista de inodos libres (actualizando el campo `SB.posPrimerInodoLibre`) y tendremos un inodo más libre en el sistema, `SB.cantInodosLibres`, y por otro lado, habrá que recorrer la estructura de enlaces del inodo para liberar **todos** aquellos **bloques de datos** que estaba ocupando, más **todos** aquellos **bloques índice** que se hubieran creado para apuntar a esos bloques.

En detalle, esta función deberá realizar las siguientes acciones:

- Leer el inodo.
- Llamar a la función auxiliar `liberar_bloques_inodo()` para liberar todos los bloques del inodo. El argumento `primerBL` que le pasamos, **valdrá 0** cuando la llamamos desde esta función, ya que si liberamos el inodo hemos de liberar también TODOS los bloques ocupados, desde el 0 hasta el último bloque de ese inodo (lo calcularemos a partir del tamaño en bytes lógico del inodo, `inodo.tamEnBytesLog`, leyendo ese dato del inodo). Hay que tener en cuenta que NO todos los **bloques lógicos** tienen porqué estar ocupados.
- A la cantidad de bloques ocupados del inodo, `inodo.numBloquesOcupados`, se le restará la cantidad de bloques liberados por la función anterior (y debería quedar a 0).
- Marcar el inodo como tipo libre y `tamEnBytesLog=0`
- Actualizar la lista enlazada de inodos libres:

<sup>1</sup> `truncar.c` no pertenece al sistema de ficheros, es para realizar testing de las funciones mientras no tengamos completada la capa de directorios.

- **Leer el superbloque**<sup>2</sup> para saber cuál es el primer inodo libre de la lista enlazada, `SB.posPrimerInodoLibre`.
- Incluir el inodo que queremos liberar en la lista de inodos libres (por el principio), actualizando el superbloque para que éste sea ahora el primero de la lista. El inodo liberado apuntará donde antes apuntaba el campo del superbloque, `SB.posPrimerInodoLibre`.



Lista de inodos libres antes y después de liberar el inodo ninodo

- En el superbloque, incrementar la cantidad de **inodos libres**, `SB.cantInodosLibres`, (la cantidad de **bloques libres** ya queda incrementada en la función `liberar_bloque()` cuando la llamamos desde `liberar_bloque_inodo()`).
- Escribir el inodo
- Escribir el superbloque
- Devolver el nº del inodo liberado

Es recomendable que la parte que libera bloques se haga con la siguiente función (ya que también la usaremos dentro de la función `mi_truncar_f()` de `ficheros.c`):

**2) `int liberar_bloques_inodo(unsigned int primerBL, struct inodo *inodo);`**

La función `liberar_bloques_inodo()` libera todos los bloques **ocupados** (con la ayuda de la función `liberar_bloque()`) a partir del bloque lógico indicado por el argumento `primerBL` (inclusive). Esta función también la usaremos dentro de la función `mi_truncar_f()`, pero

<sup>2</sup> Si lo hubiérais leído al inicio de la función de `liberar_inodo()` hay que tener en cuenta que `liberar_bloques_inodo()` cambia el valor de `SB.cantBloquesLibres`

para liberar los bloques a partir de un determinado bloque lógico. Evidentemente, en nuestro caso desde `liberar_inodo()`, llamaremos a la función `liberar_bloques_inodo()` de manera que el argumento `primerBL` valga 0, dado que nos interesa liberar los bloques lógicos del inodo desde el 0, pero sólo **hasta el último bloque lógico del fichero**. Podemos calcular cuál es ese de la siguiente manera:

```
si inodo->tamEnBytesLog % BLOCKSIZE = 0 entonces
    ultimoBL := inodo->tamEnBytesLog / BLOCKSIZE - 1
si_no ultimoBL := inodo->tamEnBytesLog / BLOCKSIZE
fsi
```

Vamos liberando **los bloques que estén ocupados** con ayuda de la función `liberar_bloque()` y contabilizamos los bloques liberados. Hay que tener en cuenta que podemos tener huecos en nuestro fichero lógico, es decir, no tienen porqué estar ocupados todos los bloques lógicos anteriores ya que podemos escribir con acceso directo a cualquier posición, aunque las anteriores no estén escritas.

Hay que comprobar cuando pasemos por un bloque índice, si al eliminar un puntero concreto, no le quedan ya más punteros ocupados, puesto que **en tal caso también habría que liberar ese bloque de punteros**. Eso lo podemos averiguar utilizando la función `memcmp()` con un buffer auxiliar inicializado a 0s con `memset()`:

```
unsigned char bufAux_punteros[BLOCKSIZE]; //1024 bytes
unsigned int bloque_punteros [NPUNTEROS]; //1024 bytes
memset (bufAux_punteros, 0, BLOCKSIZE);
if (memcmp (bloque_punteros, bufAux_punteros, BLOCKSIZE)==0)
...
```

Para desarrollar `liberar_bloques_inodo()`, podemos optar tanto por una **versión iterativa** como por una **versión recursiva** ayudándonos, si nos son útiles, de las funciones ya implementadas `obtener_nRangoBL()` y `obtener_indice()`, y de las funciones auxiliares que consideremos necesario.

Esta función ha de devolver la **cantidad de bloques liberados**. La función que llame a ésta (`liberar_inodo()` o `mi_truncar_f()`) ya se encargará de disminuir la cantidad de bloques ocupados en el inodo en base a esa cantidad, y también de actualizar el `ctime` y escribir el inodo actualizado.

Ejemplo de algoritmo iterativo (a optimizar)<sup>3</sup>:

```
funcion liberar_bloques_inodo(primerBL:unsigned int, *inodo:struct inodo) devolver liberados:ent
// libera los bloques de datos e índices iterando desde el primer bloque lógico a liberar hasta el último
// por tanto explora las ramificaciones de punteros desde las hojas hacia las raíces en el inodo

var
```

<sup>3</sup> En este algoritmo habría que minimizar los accesos a disco con `bwrite()` y también saltar la exploración de los bloques lógicos de cualquier ramificación de punteros cuando la raíz de ésta es igual a 0. Es optativo mejorarlo, y en tal caso se obtendrá puntuación extra.

```
nivel_punteros, indice, ptr, nBL, ultimoBL: unsigned ent
nRangoBL: ent
bloques_punteros [3] [NPUNTEROS]: unsigned ent //array de bloques de punteros
bufAux_punteros [BLOCKSIZE]: unsigned char
ptr_nivel [3]: ent //punteros a bloques de punteros de cada nivel
indices[3]: ent //indices de cada nivel
liberados: ent // n° de bloques liberados
fvar

liberados:=0
si inodo->tamEnBytesLog = 0 entonces devolver 0 fsi // el fichero está vacío
//obtenemos el último bloque lógico del inodo
si inodo->tamEnBytesLog % BLOCKSIZE = 0 entonces
    ultimoBL := inodo->tamEnBytesLog / BLOCKSIZE - 1
si_no ultimoBL := inodo->tamEnBytesLog / BLOCKSIZE
fsi
memset(bufAux_punteros, 0, BLOCKSIZE)
ptr:= 0

para nBL := primerBL hasta nBL = ultimoBL paso 1 hacer //recorrido BLs
    nRangoBL := obtener_nRangoBL(inodo, nBL, &ptr) //0:D, 1:I0, 2:I1, 3:I2
    si nRangoBL < 0 entonces devolver ERROR fsi
    nivel_punteros := nRangoBL //el nivel_punteros +alto cuelga del inodo

    mientras (ptr > 0 && nivel_punteros > 0) hacer //cuelgan bloques de punteros
        indice := obtener_indice(nBL, nivel_punteros)
        si indice=0 o nBL=primerBL entonces
            //solo leemos del dispositivo si no está ya cargado previamente en un buffer
            bread(ptr, bloques_punteros[nivel_punteros - 1])
        fsi
        ptr_nivel[nivel_punteros - 1] := ptr
        indices[nivel_punteros - 1] := indice
        ptr := bloques_punteros[nivel_punteros - 1][indice]
        nivel_punteros--
    fmientras

    si ptr > 0 entonces //si existe bloque de datos
        liberar_bloque(ptr)
        liberados++
        si nRangoBL = 0 entonces //es un puntero Directo
            inodo->punterosDirectos[nBL]:= 0
        si_no
            nivel_punteros:= 1
            mientras nivel_punteros <= nRangoBL hacer
                indice := indices[nivel_punteros - 1]
                bloques_punteros[nivel_punteros - 1][indice] := 0
                ptr := ptr_nivel [nivel_punteros - 1]
                si memcmp( bloques_punteros[nivel_punteros - 1], bufAux_punteros, BLOCKSIZE) = 0
                    entonces
                        //No cuelgan más bloques ocupados, hay que liberar el bloque de punteros
                        liberar_bloque(ptr)
```

```
    liberados++
    //Incluir mejora para saltar los bloques que no sea necesario explorar !!!
    ...
    si nivel_punteros = nRangoBL entonces
        inodo->punterosIndirectos[nRangoBL - 1] := 0
    fsi
    nivel_punteros++
si_no //escribimos en el dispositivo el bloque de punteros modificado
    bwrite(ptr, bloques_punteros[nivel_punteros - 1])
    // hemos de salir del bucle ya que no será necesario liberar los bloques de niveles
    // superiores de los que cuelga
    nivel_punteros := nRangoBL + 1
fsi
fmientras
    fsi
fpara
devolver liberados
ffuncion
```

Hay que incluir la declaración de las funciones anteriores en el fichero `ficheros_basico.h`.

A continuación viene una función de la capa de ficheros que también hace uso de `liberar_bloques_inodo()` pero liberando desde un determinado `BL` hasta el final:

3) `int mi_truncar_f(unsigned int ninodo, unsigned int nbytes);`

Trunca un fichero/directorio (correspondiente al nº de inodo, `ninodo`, pasado como argumento) a los bytes indicados como `nbytes`, liberando los bloques necesarios.

En nuestro sistema de ficheros, esta función será llamada desde la función `mi_unlink()` de la capa de directorios, la cuál a su vez será llamada desde el programa `mi_rm.c`, y nos servirá para eliminar una entrada de un directorio.

Hay que comprobar que el inodo tenga permisos de escritura.

No se puede truncar más allá del tamaño en bytes lógicos del fichero/directorio.

Nos basaremos en la función `liberar_bloques_inodo()`. Para saber que nº de bloque lógico le hemos de pasar como primer bloque lógico a liberar:

```
si nbytes % BLOCKSIZE = 0 entonces primerBL := nbytes/BLOCKSIZE
si_no primerBL := nbytes/BLOCKSIZE + 1
```

Actualizar *mtime*, *ctime*, el tamaño en bytes lógicos del inodo, *tamEnBytesLog* (**pasará a ser igual a *nbytes***) y el número de bloques ocupados del inodo, *numBloquesOcupados* (habrá que restarle los liberados).

Devolver la cantidad de bloques liberados.

A continuación habrá que hacer un programita ficticio, **truncar.c**, para poder disponer desde consola de un comando que ejecute las funciones *liberar\_inodo()* o *mi\_truncar\_f()* hasta que dispongamos de las funciones de capas superiores que llamen a ésta de forma natural:

### truncar.c

Sintaxis: **truncar <nombre\_dispositivo> <ninodo> <nbytes>**

Pasos a realizar:

- Validación de sintaxis
- montar dispositivo virtual
- si *nbytes*= 0 *liberar\_inodo()* si no *mi\_truncar\_f()* fsi
- desmontar dispositivo virtual

*nbytes* puede ser 0 (simularía la llamada de *mi\_truncar\_f()* desde *liberar\_inodo()*), o bien un valor que al menos deje 1 byte escrito en el inodo ya que si no el campo *tamEnBytesLog* se quedará incoherente. Cuando esté la práctica completa nunca indicaremos *nbytes* desde consola, es sólo ahora para poder testear nuestras funciones.

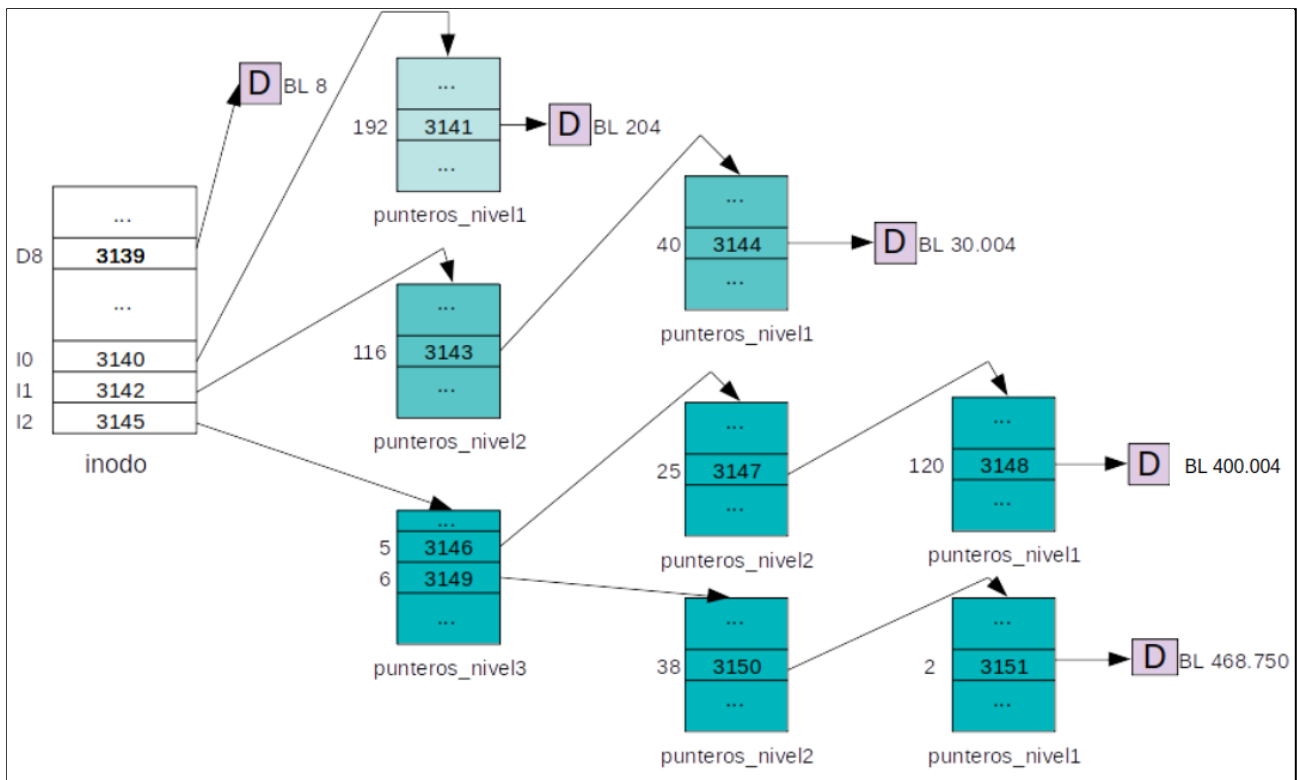
No sólo hay que comprobar que el tamaño del inodo pasa a ser *nbytes* sino también que se libera la cantidad de bloques correcta (han de quedar con valor correcto los campos *numBloquesOcupados* del inodo y *cantBloquesLibres* del superbloque):

- La *cantBloquesLibres* del SB sólo la debería decrementar la función *reservar\_bloque()*, y solo lo debería incrementar la función *liberar\_bloques()*.
- El *numBloquesOcupados* del inodo sólo lo debería incrementar la función *traducir\_bloque\_inodo()* con *reservar*=1, y sólo lo debería decrementar la función *mi\_truncar\_f()* o *liberar\_inodo()*.

Al finalizar llamada a *mi\_stat\_f()* para mostrar al menos el *tamEnBytesLog* y *numBloquesOcupados* para comprobar que son correctos.

## Tests de prueba

- Ejecución de **script\_truncar.sh** que escribe "123456789" en diferentes *offsets* de un mismo inodo ( $9.000 \subset \text{BL } 8$ ,  $209.000 \subset \text{BL } 204$ ,  $30.725.000 \subset \text{BL } 30.004$ ,  $409.605.000 \subset \text{BL } 400.004$ ,  $480.000.000 \subset \text{BL } 468.750$ ) y luego elimina el inodo con todos sus bloques



Punteros inodo	Bloques lógicos	Bytes lógicos
punterosDirectos [0] ... [11]	BL 0 ... BL 11	0 ... 12.287
punterosIndirectos[0]	BL 12 ... BL 267	12.288 ... 274.431
punterosIndirectos[1]	BL 268 ... BL 65.803	273.432 ... 67.383.295
punterosIndirectos[2]	BL 65.804 ... 16.843.019	67.383.296 ... 17.247.252.479

Tabla de rangos para BLOCKSIZE=1024

```

$ ./script_truncar.sh
$ ./mi_mkfs disco 100000
    
```



**#inicializamos el sistema de ficheros con 100.000 bloques**

#####

**\$ ./leer\_sf disco**

**#mostramos solo el SB**

DATOS DEL SUPERBLOQUE

posPrimerBloqueMB = 1

posUltimoBloqueMB = 13

posPrimerBloqueAI = 14

posUltimoBloqueAI = 3138

posPrimerBloqueDatos = 3139

posUltimoBloqueDatos = 99999

posInodoRaiz = 0

posPrimerInodoLibre = 1

cantBloquesLibres = 96861

cantInodosLibres = 24999

totBloques = 100000

totInodos = 25000

#####

**\$ ./escribir disco 123456789 0**

longitud texto: 9

Nº inodo reservado: 1

offset: 9000

[traducir\_bloque\_inodo()→ inodo.punterosDirectos[8] = 3139 (reservado BF 3139 para BL 8)]

Bytes escritos: 9

stat.tamEnBytesLog=9009

stat.numBloquesOcupados=1

Nº inodo reservado: 1

offset: 209000

[traducir\_bloque\_inodo()→ inodo.punterosIndirectos[0] = 3140 (reservado BF 3140 para punteros\_nivel1)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [192] = 3141 (reservado BF 3141 para BL 204)]

Bytes escritos: 9

stat.tamEnBytesLog=209009

stat.numBloquesOcupados=3

Nº inodo reservado: 1

offset: 30725000

[traducir\_bloque\_inodo()→ inodo.punterosIndirectos[1] = 3142 (reservado BF 3142 para punteros\_nivel2)]

[traducir\_bloque\_inodo()→ punteros\_nivel2 [116] = 3143 (reservado BF 3143 para punteros\_nivel1)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [40] = 3144 (reservado BF 3144 para BL 30004)]

Bytes escritos: 9

stat.tamEnBytesLog=30725009



```
stat.numBloquesOcupados=6
```

```
Nº inodo reservado: 1
```

```
offset: 409605000
```

```
[traducir_bloque_inodo()→ inodo.punterosIndirectos[2] = 3145 (reservado BF 3145 para  
punteros_nivel3)]
```

```
[traducir_bloque_inodo()→ punteros_nivel3 [5] = 3146 (reservado BF 3146 para punteros_nivel2)]
```

```
[traducir_bloque_inodo()→ punteros_nivel2 [25] = 3147 (reservado BF 3147 para  
punteros_nivel1)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [120] = 3148 (reservado BF 3148 para BL 400004)]
```

```
Bytes escritos: 9
```

```
stat.tamEnBytesLog=409605009
```

```
stat.numBloquesOcupados=10
```

```
Nº inodo reservado: 1
```

```
offset: 480000000
```

```
[traducir_bloque_inodo()→ punteros_nivel3 [6] = 3149 (reservado BF 3149 para punteros_nivel2)]
```

```
[traducir_bloque_inodo()→ punteros_nivel2 [38] = 3150 (reservado BF 3150 para  
punteros_nivel1)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [2] = 3151 (reservado BF 3151 para BL 468750)]
```

```
Bytes escritos: 9
```

```
stat.tamEnBytesLog=480000009
```

```
stat.numBloquesOcupados=13
```

```
#####
```

```
$ ./leer_sf disco
```

```
#mostramos solo el SB
```

```
DATOS DEL SUPERBLOQUE
```

```
...
```

```
posPrimerInodoLibre = 2
```

```
cantBloquesLibres = 96848
```

```
cantInodosLibres = 24998
```

```
...
```

```
#####
```

```
$ ./truncar disco 1 0
```

```
[liberar_bloques_inodo()→ primer BL: 0, último BL: 468750]
```

```
[liberar_bloques_inodo()→ liberado BF 3139 de datos para BL 8]
```

```
[liberar_bloques_inodo()→ liberado BF 3141 de datos para BL 204]
```

```
[liberar_bloques_inodo()→ liberado BF 3140 de punteros_nivel1 correspondiente al BL 204]
```

```
[liberar_bloques_inodo()→ liberado BF 3144 de datos para BL 30004]
```

```
[liberar_bloques_inodo()→ liberado BF 3143 de punteros_nivel1 correspondiente al BL 30004]
```

```
[liberar_bloques_inodo()→ liberado BF 3142 de punteros_nivel2 correspondiente al BL 30004]
```

```
[liberar_bloques_inodo()→ liberado BF 3148 de datos para BL 400004]
```

```
[liberar_bloques_inodo()→ liberado BF 3147 de punteros_nivel1 correspondiente al BL 400004]
```

```
[liberar_bloques_inodo()→ liberado BF 3146 de punteros_nivel2 correspondiente al BL 400004]
```

```
[liberar_bloques_inodo()→ liberado BF 3151 de datos para BL 468750]
```

# Sistema de ficheros

## Nivel 6

**ficheros\_basico.c** {liberar\_inodo(),  
liberar\_bloques\_inodo()}, **ficheros.c**  
{truncar\_f()}, **truncar.c**

```
[liberar_bloques_inodo()→ liberado BF 3150 de punteros_nivel1 correspondiente al BL 468750]  
[liberar_bloques_inodo()→ liberado BF 3149 de punteros_nivel2 correspondiente al BL 468750]  
[liberar_bloques_inodo()→ liberado BF 3145 de punteros_nivel3 correspondiente al BL 468750]  
[liberar_bloques_inodo()→ total bloques liberados: 13]
```

```
real    0m0,659s  
user    0m0,363s  
sys     0m0,295s
```

DATOS INODO 1:

```
tipo=l  
permisos=6  
atime: Tue 2021-03-23 17:51:38  
ctime: Tue 2021-03-23 17:51:38  
mtime: Tue 2021-03-23 17:51:38  
nlinks=1  
tamEnBytesLog=0  
numBloquesOcupados=0
```

```
real    0m0,111s  
user    0m0,055s  
sys     0m0,055s
```

```
#####
```

```
$ ./leer_sf disco
```

```
#mostramos solo el SB
```

DATOS DEL SUPERBLOQUE

```
...  
posPrimerInodoLibre = 1  
cantBloquesLibres = 96861  
cantInodosLibres = 24999  
...
```

- Ejecución de **script\_truncar\_parcial.sh** que escribe "123456789" en diferentes offsets de un mismo inodo y luego va haciendo truncamientos sucesivos desde diferentes bytes, que van recortando el tamaño en bytes lógico del fichero.

```
$ ./script_truncar_parcial.sh
```

```
$ ./mi_mkfs disco 100000
```

```
#inicializamos el sistema de ficheros con 100.000 bloques
```

```
$ ./leer_sf disco
```

DATOS DEL SUPERBLOQUE

```
...
posPrimerInodoLibre = 1
cantBloquesLibres = 96861
cantInodosLibres = 24999
...

$ ./escribir disco 123456789 0
longitud texto: 9

Nº inodo reservado: 1
offset: 9000
[traducir_bloque_inodo()→ inodo.punterosDirectos[8] = 3139 (reservado BF 3139 para BL 8)]
Bytes escritos: 9
stat.tamEnBytesLog=9009
stat.numBloquesOcupados=1

Nº inodo reservado: 1
offset: 209000
[traducir_bloque_inodo()→ inodo.punterosIndirectos[0] = 3140 (reservado BF 3140 para
punteros_nivel1)]
[traducir_bloque_inodo()→ punteros_nivel1 [192] = 3141 (reservado BF 3141 para BL 204)]
Bytes escritos: 9
stat.tamEnBytesLog=209009
stat.numBloquesOcupados=3

Nº inodo reservado: 1
offset: 30725000
[traducir_bloque_inodo()→ inodo.punterosIndirectos[1] = 3142 (reservado BF 3142 para
punteros_nivel2)]
[traducir_bloque_inodo()→ punteros_nivel2 [116] = 3143 (reservado BF 3143 para
punteros_nivel1)]
[traducir_bloque_inodo()→ punteros_nivel1 [40] = 3144 (reservado BF 3144 para BL 30004)]
Bytes escritos: 9
stat.tamEnBytesLog=30725009
stat.numBloquesOcupados=6

Nº inodo reservado: 1
offset: 409605000
[traducir_bloque_inodo()→ inodo.punterosIndirectos[2] = 3145 (reservado BF 3145 para
punteros_nivel3)]
[traducir_bloque_inodo()→ punteros_nivel3 [5] = 3146 (reservado BF 3146 para punteros_nivel2)]
[traducir_bloque_inodo()→ punteros_nivel2 [25] = 3147 (reservado BF 3147 para
punteros_nivel1)]
[traducir_bloque_inodo()→ punteros_nivel1 [120] = 3148 (reservado BF 3148 para BL 400004)]
Bytes escritos: 9
stat.tamEnBytesLog=409605009
stat.numBloquesOcupados=10
```

```
Nº inodo reservado: 1
offset: 480000000
[traducir_bloque_inodo()→ punteros_nivel3 [6] = 3149 (reservado BF 3149 para punteros_nivel2)]
[traducir_bloque_inodo()→ punteros_nivel2 [38] = 3150 (reservado BF 3150 para
punteros_nivel1)]
[traducir_bloque_inodo()→ punteros_nivel1 [2] = 3151 (reservado BF 3151 para BL 468750)]
Bytes escritos: 9
stat.tamEnBytesLog=480000009
stat.numBloquesOcupados=13
```

**\$ ./leer\_sf disco**

DATOS DEL SUPERBLOQUE

```
...
posPrimerInodoLibre = 2
cantBloquesLibres = 96848
cantInodosLibres = 24998
...
```

**\$ ./truncar disco 1 409605001<sup>4</sup>**

```
[liberar_bloques_inodo()→ primer BL: 400005, último BL: 468750]
[liberar_bloques_inodo()→ liberado BF 3151 de datos para BL 468750]
[liberar_bloques_inodo()→ liberado BF 3150 de punteros_nivel1 correspondiente al BL 468750]
[liberar_bloques_inodo()→ liberado BF 3149 de punteros_nivel2 correspondiente al BL 468750]
[liberar_bloques_inodo()→ total bloques liberados: 3]
liberados: 3
```

DATOS INODO 1:

```
tipo=f
permisos=6
atime: Tue 2021-03-23 17:58:50
ctime: Tue 2021-03-23 17:58:50
mtime: Tue 2021-03-23 17:58:50
nlinks=1
tamEnBytesLog=409605001
numBloquesOcupados=10
```

**\$ ./leer\_sf disco**

DATOS DEL SUPERBLOQUE

```
...
posPrimerInodoLibre = 2
cantBloquesLibres = 96851
cantInodosLibres = 24998
...
```

<sup>4</sup> byte 409.605.001  $\subset$  BL 400.004 (hay que preservarlo ya que aún contiene otros bytes y empezar a liberar a partir del siguiente)

**\$ ./truncar disco 1 30725003<sup>5</sup>**

[liberar\_bloques\_inodo()→ primer BL: 30005, último BL: 400004]

[liberar\_bloques\_inodo()→ liberado BF 3148 de datos para BL 400004]

[liberar\_bloques\_inodo()→ liberado BF 3147 de punteros\_nivel1 correspondiente al BL 400004]

[liberar\_bloques\_inodo()→ liberado BF 3146 de punteros\_nivel2 correspondiente al BL 400004]

[liberar\_bloques\_inodo()→ liberado BF 3145 de punteros\_nivel3 correspondiente al BL 400004]

[liberar\_bloques\_inodo()→ total bloques liberados: 4]

liberados: 4

DATOS INODO 1:

tipo=f

permisos=6

atime: Tue 2021-03-23 17:58:50

ctime: Tue 2021-03-23 17:58:50

mtime: Tue 2021-03-23 17:58:50

nlinks=1

tamEnBytesLog=30725003

numBloquesOcupados=6

**\$ ./leer\_sf disco**

DATOS DEL SUPERBLOQUE

...

posPrimerInodoLibre = 2

cantBloquesLibres = 96855

cantInodosLibres = 24998

...

**\$ ./truncar disco 1 209008<sup>6</sup>**

[liberar\_bloques\_inodo()→ primer BL: 205, último BL: 30004]

[liberar\_bloques\_inodo()→ liberado BF 3144 de datos para BL 30004]

[liberar\_bloques\_inodo()→ liberado BF 3143 de punteros\_nivel1 correspondiente al BL 30004]

[liberar\_bloques\_inodo()→ liberado BF 3142 de punteros\_nivel2 correspondiente al BL 30004]

[liberar\_bloques\_inodo()→ total bloques liberados: 3]

liberados: 3

DATOS INODO 1:

tipo=f

permisos=6

atime: Tue 2021-03-23 17:58:50

ctime: Tue 2021-03-23 17:58:50

mtime: Tue 2021-03-23 17:58:50

<sup>5</sup> byte 30.725.003  $\subset$  BL 30.004 (hay que preservarlo ya que aún contiene otros bytes y empezar a liberar a partir del siguiente)

<sup>6</sup> byte 209.008  $\subset$  BL 204 (hay que preservarlo ya que aún contiene otros bytes y empezar a liberar a partir del siguiente)

```
nlinks=1
tamEnBytesLog=209008
numBloquesOcupados=3

$ ./leer_sf disco

DATOS DEL SUPERBLOQUE
...
posPrimerInodoLibre = 2
cantBloquesLibres = 96858
cantInodosLibres = 24998
...

$ ./truncar disco 1 90057
[liberar_bloques_inodo()→ primer BL: 9, último BL: 204]
[liberar_bloques_inodo()→ liberado BF 3141 de datos para BL 204]
[liberar_bloques_inodo()→ liberado BF 3140 de punteros_nivel1 correspondiente al BL 204]
[liberar_bloques_inodo()→ total bloques liberados: 2]
liberados: 2

DATOS INODO 1:
tipo=f
permisos=6
atime: Tue 2021-03-23 17:58:50
ctime: Tue 2021-03-23 17:58:50
mtime: Tue 2021-03-23 17:58:50
nlinks=1
tamEnBytesLog=9005
numBloquesOcupados=1

$ ./leer_sf disco

DATOS DEL SUPERBLOQUE
...
posPrimerInodoLibre = 2
cantBloquesLibres = 96860
cantInodosLibres = 24998
...

$ ./truncar disco 1 0
[liberar_bloques_inodo()→ primer BL: 0, último BL: 8]
[liberar_bloques_inodo()→ liberado BF 3139 de datos para BL 8]
[liberar_bloques_inodo()→ total bloques liberados: 1]

DATOS INODO 1:
```

<sup>7</sup> byte 9.005 < BL 8 (hay que preservarlo ya que aún contiene otros bytes y empezar a liberar a partir del siguiente)

# Sistema de ficheros

## Nivel 6

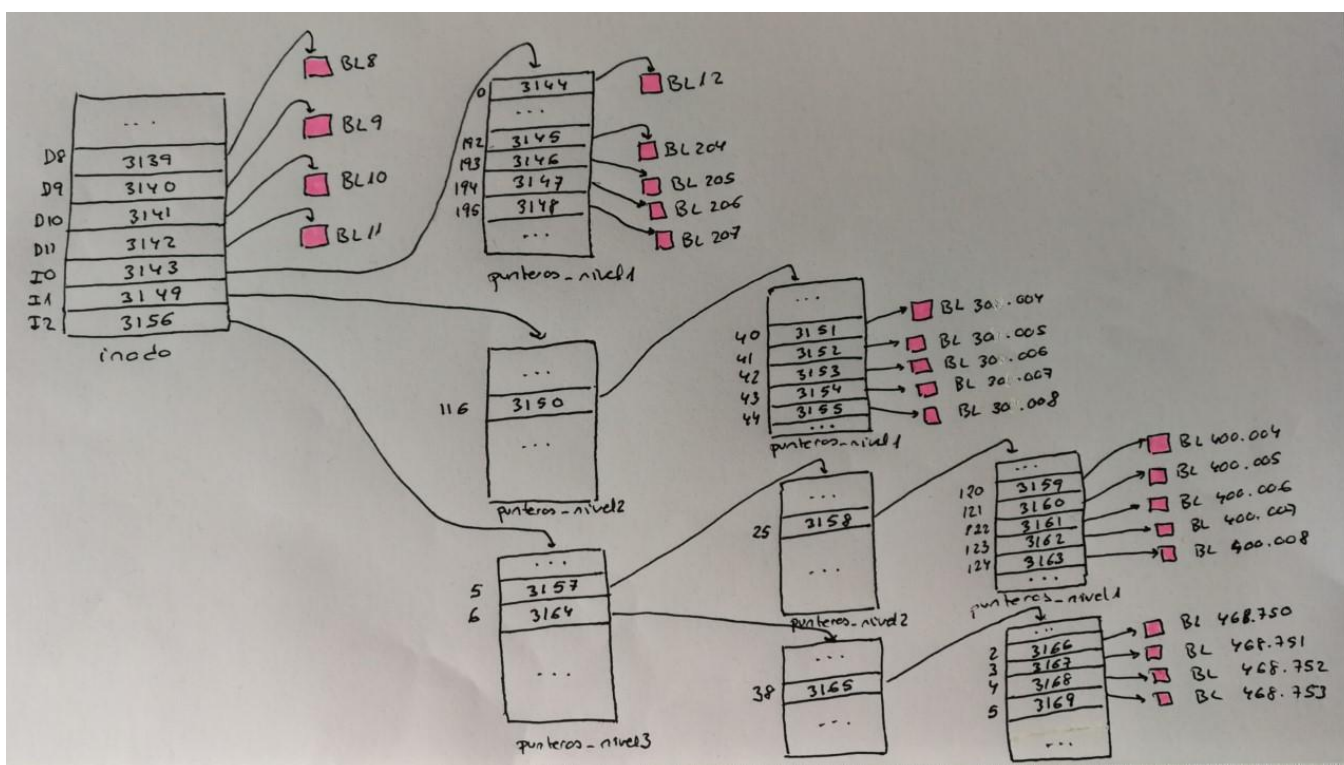
`ficheros_basico.c` {`liberar_inodo()`,  
`liberar_bloques_inodo()`}, `ficheros.c`  
{`truncar_f()`}, `truncar.c`

```
tipo=1
permisos=6
atime: Tue 2021-03-23 17:58:50
ctime: Tue 2021-03-23 17:58:50
mtime: Tue 2021-03-23 17:58:50
nlinks=1
tamEnBytesLog=0
numBloquesOcupados=0
$ ./leer_sf disco
```

### DATOS DEL SUPERBLOQUE

```
...
posPrimerInodoLibre = 1
cantBloquesLibres = 96861
cantInodosLibres = 24999
...
```

- Ejecución de `script_truncar2.sh` que escribe el `texto2.txt`, que ocupa varios bloques, en diferentes offsets de un mismo inodo y luego libera el inodo y todos sus bloques



```
$ ./script_truncar2.sh
$ ./mi_mkfs disco 100000
```



**#inicializamos el sistema de ficheros con 100.000 bloques**

**\$ ./leer\_sf disco**

**#mostramos solo el SB**

DATOS DEL SUPERBLOQUE

...

posPrimerInodoLibre = 1

cantBloquesLibres = 96861

cantInodosLibres = 24999

...

**\$ ./escribir disco "\$(cat texto2.txt)" 0**

Offsets: 9.000, 209.000, 30.725.000, 409.605.000, 480.000.000

longitud texto: 3751

Nº inodo reservado: 1

offset: 9000

[traducir\_bloque\_inodo()→ inodo.punterosDirectos[8] = 3139 (reservado BF 3139 para BL 8)]

[traducir\_bloque\_inodo()→ inodo.punterosDirectos[9] = 3140 (reservado BF 3140 para BL 9)]

[traducir\_bloque\_inodo()→ inodo.punterosDirectos[10] = 3141 (reservado BF 3141 para BL 10)]

[traducir\_bloque\_inodo()→ inodo.punterosDirectos[11] = 3142 (reservado BF 3142 para BL 11)]

[traducir\_bloque\_inodo()→ inodo.punterosIndirectos[0] = 3143 (reservado BF 3143 para punteros\_nivel1)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [0] = 3144 (reservado BF 3144 para BL 12)]

Bytes escritos: 3751

stat.tamEnBytesLog=12751

stat.numBloquesOcupados=6

Nº inodo reservado: 1

offset: 209000

[traducir\_bloque\_inodo()→ punteros\_nivel1 [192] = 3145 (reservado BF 3145 para BL 204)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [193] = 3146 (reservado BF 3146 para BL 205)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [194] = 3147 (reservado BF 3147 para BL 206)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [195] = 3148 (reservado BF 3148 para BL 207)]

Bytes escritos: 3751

stat.tamEnBytesLog=212751

stat.numBloquesOcupados=10

Nº inodo reservado: 1

offset: 30725000

[traducir\_bloque\_inodo()→ inodo.punterosIndirectos[1] = 3149 (reservado BF 3149 para punteros\_nivel2)]

[traducir\_bloque\_inodo()→ punteros\_nivel2 [116] = 3150 (reservado BF 3150 para punteros\_nivel1)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [40] = 3151 (reservado BF 3151 para BL 30004)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [41] = 3152 (reservado BF 3152 para BL 30005)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [42] = 3153 (reservado BF 3153 para BL 30006)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [43] = 3154 (reservado BF 3154 para BL 30007)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [44] = 3155 (reservado BF 3155 para BL 30008)]

Bytes escritos: 3751

stat.tamEnBytesLog=30728751

stat.numBloquesOcupados=17

Nº inodo reservado: 1

offset: 409605000

[traducir\_bloque\_inodo()→ inodo.punterosIndirectos[2] = 3156 (reservado BF 3156 para punteros\_nivel3)]

[traducir\_bloque\_inodo()→ punteros\_nivel3 [5] = 3157 (reservado BF 3157 para punteros\_nivel2)]

[traducir\_bloque\_inodo()→ punteros\_nivel2 [25] = 3158 (reservado BF 3158 para punteros\_nivel1)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [120] = 3159 (reservado BF 3159 para BL 400004)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [121] = 3160 (reservado BF 3160 para BL 400005)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [122] = 3161 (reservado BF 3161 para BL 400006)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [123] = 3162 (reservado BF 3162 para BL 400007)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [124] = 3163 (reservado BF 3163 para BL 400008)]

Bytes escritos: 3751

stat.tamEnBytesLog=409608751

stat.numBloquesOcupados=25

Nº inodo reservado: 1

offset: 480000000

[traducir\_bloque\_inodo()→ punteros\_nivel3 [6] = 3164 (reservado BF 3164 para punteros\_nivel2)]

[traducir\_bloque\_inodo()→ punteros\_nivel2 [38] = 3165 (reservado BF 3165 para punteros\_nivel1)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [2] = 3166 (reservado BF 3166 para BL 468750)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [3] = 3167 (reservado BF 3167 para BL 468751)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [4] = 3168 (reservado BF 3168 para BL 468752)]

[traducir\_bloque\_inodo()→ punteros\_nivel1 [5] = 3169 (reservado BF 3169 para BL 468753)]

Bytes escritos: 3751

stat.tamEnBytesLog=480003751

stat.numBloquesOcupados=31

### \$ ./truncar disco 1 0

[liberar\_bloques\_inodo()→ primer BL: 0, último BL: 468753]

[liberar\_bloques\_inodo()→ liberado BF 3139 de datos para BL 8]

[liberar\_bloques\_inodo()→ liberado BF 3140 de datos para BL 9]

[liberar\_bloques\_inodo()→ liberado BF 3141 de datos para BL 10]

[liberar\_bloques\_inodo()→ liberado BF 3142 de datos para BL 11]

[liberar\_bloques\_inodo()→ liberado BF 3144 de datos para BL 12]

[liberar\_bloques\_inodo()→ liberado BF 3145 de datos para BL 204]

[liberar\_bloques\_inodo()→ liberado BF 3146 de datos para BL 205]

[liberar\_bloques\_inodo()→ liberado BF 3147 de datos para BL 206]

[liberar\_bloques\_inodo()→ liberado BF 3148 de datos para BL 207]

[liberar\_bloques\_inodo()→ liberado BF 3143 de punteros\_nivel1 correspondiente al BL 207]

[liberar\_bloques\_inodo()→ liberado BF 3151 de datos para BL 30004]

[liberar\_bloques\_inodo()→ liberado BF 3152 de datos para BL 30005]

```
[liberar_bloques_inodo()→ liberado BF 3153 de datos para BL 30006]
[liberar_bloques_inodo()→ liberado BF 3154 de datos para BL 30007]
[liberar_bloques_inodo()→ liberado BF 3155 de datos para BL 30008]
[liberar_bloques_inodo()→ liberado BF 3150 de punteros_nivel1 correspondiente al BL 30008]
[liberar_bloques_inodo()→ liberado BF 3149 de punteros_nivel2 correspondiente al BL 30008]
[liberar_bloques_inodo()→ liberado BF 3159 de datos para BL 400004]
[liberar_bloques_inodo()→ liberado BF 3160 de datos para BL 400005]
[liberar_bloques_inodo()→ liberado BF 3161 de datos para BL 400006]
[liberar_bloques_inodo()→ liberado BF 3162 de datos para BL 400007]
[liberar_bloques_inodo()→ liberado BF 3163 de datos para BL 400008]
[liberar_bloques_inodo()→ liberado BF 3158 de punteros_nivel1 correspondiente al BL 400008]
[liberar_bloques_inodo()→ liberado BF 3157 de punteros_nivel2 correspondiente al BL 400008]
[liberar_bloques_inodo()→ liberado BF 3166 de datos para BL 468750]
[liberar_bloques_inodo()→ liberado BF 3167 de datos para BL 468751]
[liberar_bloques_inodo()→ liberado BF 3168 de datos para BL 468752]
[liberar_bloques_inodo()→ liberado BF 3169 de datos para BL 468753]
[liberar_bloques_inodo()→ liberado BF 3165 de punteros_nivel1 correspondiente al BL 468753]
[liberar_bloques_inodo()→ liberado BF 3164 de punteros_nivel2 correspondiente al BL 468753]
[liberar_bloques_inodo()→ liberado BF 3156 de punteros_nivel3 correspondiente al BL 468753]
[liberar_bloques_inodo()→ total bloques liberados: 31]
```

DATOS INODO 1:

tipo=l

permisos=6

atime: Tue 2021-03-23 18:16:21

ctime: Tue 2021-03-23 18:16:21

mtime: Tue 2021-03-23 18:16:21

nlinks=1

**tamEnBytesLog=0**

**numBloquesOcupados=0**

real 0m0,116s

user 0m0,060s

sys 0m0,056s

- Ejecución de **script\_truncar\_parcial2.sh** que escribe el **texto2.txt**, que ocupa varios bloques, en diferentes offsets de un mismo inodo y luego va haciendo truncamientos sucesivos que van recortando el tamaño en bytes lógico del fichero.
  - El primer truncamiento es en el byte 409.605.001  $\subset$  BL 400.004, y quedan más bytes escritos en ese BL, por tanto no se puede eliminar y supondrá liberar los BL ocupados desde el 400.005 al 468.753
  - El segundo truncamiento es en el byte 30.725.003  $\subset$  BL 30.004, y quedan más bytes escritos en ese BL, por tanto no se puede eliminar y supondrá liberar los BL ocupados desde el 30.005 al 400.004

- El tercer truncamiento es en el byte 209.008  $\subset$  BL 204, y quedan más bytes escritos en ese BL, por tanto no se puede eliminar y supondrá liberar los BL ocupados desde el 205 al 30.004
- El cuarto truncamiento es en el byte 9.005  $\subset$  BL 8, y quedan más bytes escritos en ese BL, por tanto no se puede eliminar y supondrá liberar los BL ocupados desde el 9 al 204
- El último truncamiento es en el byte 0 y eso implica liberar el inodo y los bloques ocupados restantes (BL 8)

```
$ ./script_truncar_parcial2.sh
$ ./mi_mkfs disco 100000
#inicializamos el sistema de ficheros con 100.000 bloques
$ ./leer_sf disco

DATOS DEL SUPERBLOQUE
...
posPrimerInodoLibre = 1
cantBloquesLibres = 96861
cantInodosLibres = 24999
...

$ ./escribir disco "$(cat texto2.txt)" 0
Offsets: 9.000, 209.000, 30.725.000, 409.605.000, 480.000.000
longitud texto: 3751

Nº inodo reservado: 1
offset: 9000
[traducir_bloque_inodo()→ inodo.punterosDirectos[8] = 3139 (reservado BF 3139 para BL 8)]
[traducir_bloque_inodo()→ inodo.punterosDirectos[9] = 3140 (reservado BF 3140 para BL 9)]
[traducir_bloque_inodo()→ inodo.punterosDirectos[10] = 3141 (reservado BF 3141 para BL 10)]
[traducir_bloque_inodo()→ inodo.punterosDirectos[11] = 3142 (reservado BF 3142 para BL 11)]
[traducir_bloque_inodo()→ inodo.punterosIndirectos[0] = 3143 (reservado BF 3143 para
punteros_nivel1)]
[traducir_bloque_inodo()→ punteros_nivel1 [0] = 3144 (reservado BF 3144 para BL 12)]
Bytes escritos: 3751
stat.tamEnBytesLog=12751
stat.numBloquesOcupados=6

Nº inodo reservado: 1
offset: 209000
[traducir_bloque_inodo()→ punteros_nivel1 [192] = 3145 (reservado BF 3145 para BL 204)]
[traducir_bloque_inodo()→ punteros_nivel1 [193] = 3146 (reservado BF 3146 para BL 205)]
[traducir_bloque_inodo()→ punteros_nivel1 [194] = 3147 (reservado BF 3147 para BL 206)]
[traducir_bloque_inodo()→ punteros_nivel1 [195] = 3148 (reservado BF 3148 para BL 207)]
Bytes escritos: 3751
stat.tamEnBytesLog=212751
```

```
stat.numBloquesOcupados=10
```

```
Nº inodo reservado: 1
```

```
offset: 30725000
```

```
[traducir_bloque_inodo()→ inodo.punterosIndirectos[1] = 3149 (reservado BF 3149 para  
punteros_nivel2)]
```

```
[traducir_bloque_inodo()→ punteros_nivel2 [116] = 3150 (reservado BF 3150 para  
punteros_nivel1)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [40] = 3151 (reservado BF 3151 para BL 30004)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [41] = 3152 (reservado BF 3152 para BL 30005)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [42] = 3153 (reservado BF 3153 para BL 30006)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [43] = 3154 (reservado BF 3154 para BL 30007)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [44] = 3155 (reservado BF 3155 para BL 30008)]
```

```
Bytes escritos: 3751
```

```
stat.tamEnBytesLog=30728751
```

```
stat.numBloquesOcupados=17
```

```
Nº inodo reservado: 1
```

```
offset: 409605000
```

```
[traducir_bloque_inodo()→ inodo.punterosIndirectos[2] = 3156 (reservado BF 3156 para  
punteros_nivel3)]
```

```
[traducir_bloque_inodo()→ punteros_nivel3 [5] = 3157 (reservado BF 3157 para punteros_nivel2)]
```

```
[traducir_bloque_inodo()→ punteros_nivel2 [25] = 3158 (reservado BF 3158 para  
punteros_nivel1)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [120] = 3159 (reservado BF 3159 para BL 400004)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [121] = 3160 (reservado BF 3160 para BL 400005)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [122] = 3161 (reservado BF 3161 para BL 400006)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [123] = 3162 (reservado BF 3162 para BL 400007)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [124] = 3163 (reservado BF 3163 para BL 400008)]
```

```
Bytes escritos: 3751
```

```
stat.tamEnBytesLog=409608751
```

```
stat.numBloquesOcupados=25
```

```
Nº inodo reservado: 1
```

```
offset: 480000000
```

```
[traducir_bloque_inodo()→ punteros_nivel3 [6] = 3164 (reservado BF 3164 para punteros_nivel2)]
```

```
[traducir_bloque_inodo()→ punteros_nivel2 [38] = 3165 (reservado BF 3165 para  
punteros_nivel1)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [2] = 3166 (reservado BF 3166 para BL 468750)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [3] = 3167 (reservado BF 3167 para BL 468751)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [4] = 3168 (reservado BF 3168 para BL 468752)]
```

```
[traducir_bloque_inodo()→ punteros_nivel1 [5] = 3169 (reservado BF 3169 para BL 468753)]
```

```
Bytes escritos: 3751
```

```
stat.tamEnBytesLog=480003751
```

```
stat.numBloquesOcupados=31
```

```
$ ./leer_sf disco
```

### DATOS DEL SUPERBLOQUE

```
...
posPrimerInodoLibre = 2
cantBloquesLibres = 96830
cantInodosLibres = 24998
...
```

### \$ ./truncar disco 1 409605001

```
[liberar_bloques_inodo()→ primer BL: 400005, último BL: 468753]
[liberar_bloques_inodo()→ liberado BF 3160 de datos para BL 400005]
[liberar_bloques_inodo()→ liberado BF 3161 de datos para BL 400006]
[liberar_bloques_inodo()→ liberado BF 3162 de datos para BL 400007]
[liberar_bloques_inodo()→ liberado BF 3163 de datos para BL 400008]
[liberar_bloques_inodo()→ liberado BF 3166 de datos para BL 468750]
[liberar_bloques_inodo()→ liberado BF 3167 de datos para BL 468751]
[liberar_bloques_inodo()→ liberado BF 3168 de datos para BL 468752]
[liberar_bloques_inodo()→ liberado BF 3169 de datos para BL 468753]
[liberar_bloques_inodo()→ liberado BF 3165 de punteros_nivel1 correspondiente al BL 468753]
[liberar_bloques_inodo()→ liberado BF 3164 de punteros_nivel2 correspondiente al BL 468753]
[liberar_bloques_inodo()→ total bloques liberados: 10]
liberados: 10
```

### DATOS INODO 1:

```
tipo=f
permisos=6
atime: Tue 2021-03-23 18:28:02
ctime: Tue 2021-03-23 18:28:02
mtime: Tue 2021-03-23 18:28:02
nlinks=1
tamEnBytesLog=409605001
numBloquesOcupados=21
```

### \$ ./leer\_sf disco

### DATOS DEL SUPERBLOQUE

```
...
posPrimerInodoLibre = 2
cantBloquesLibres = 96840
cantInodosLibres = 24998
...
```

### \$ ./truncar disco 1 30725003

```
[liberar_bloques_inodo()→ primer BL: 30005, último BL: 400004]
[liberar_bloques_inodo()→ liberado BF 3152 de datos para BL 30005]
[liberar_bloques_inodo()→ liberado BF 3153 de datos para BL 30006]
[liberar_bloques_inodo()→ liberado BF 3154 de datos para BL 30007]
[liberar_bloques_inodo()→ liberado BF 3155 de datos para BL 30008]
[liberar_bloques_inodo()→ liberado BF 3159 de datos para BL 400004]
```



```
[liberar_bloques_inodo()→ liberado BF 3158 de punteros_nivel1 correspondiente al BL 400004]
[liberar_bloques_inodo()→ liberado BF 3157 de punteros_nivel2 correspondiente al BL 400004]
[liberar_bloques_inodo()→ liberado BF 3156 de punteros_nivel3 correspondiente al BL 400004]
[liberar_bloques_inodo()→ total bloques liberados: 8]
liberados: 8
```

DATOS INODO 1:

tipo=f

permisos=6

atime: Tue 2021-03-23 18:28:02

ctime: Tue 2021-03-23 18:28:02

mtime: Tue 2021-03-23 18:28:02

nlinks=1

tamEnBytesLog=30725003

numBloquesOcupados=13

**\$ ./leer\_sf disco**

DATOS DEL SUPERBLOQUE

...

posPrimerInodoLibre = 2

cantBloquesLibres = 96848

cantInodosLibres = 24998

...

**\$ ./truncar disco 1 209008**

```
[liberar_bloques_inodo()→ primer BL: 205, último BL: 30004]
```

```
[liberar_bloques_inodo()→ liberado BF 3146 de datos para BL 205]
```

```
[liberar_bloques_inodo()→ liberado BF 3147 de datos para BL 206]
```

```
[liberar_bloques_inodo()→ liberado BF 3148 de datos para BL 207]
```

```
[liberar_bloques_inodo()→ liberado BF 3151 de datos para BL 30004]
```

```
[liberar_bloques_inodo()→ liberado BF 3150 de punteros_nivel1 correspondiente al BL 30004]
```

```
[liberar_bloques_inodo()→ liberado BF 3149 de punteros_nivel2 correspondiente al BL 30004]
```

```
[liberar_bloques_inodo()→ total bloques liberados: 6]
```

liberados: 6

DATOS INODO 1:

tipo=f

permisos=6

atime: Tue 2021-03-23 18:28:02

ctime: Tue 2021-03-23 18:28:02

mtime: Tue 2021-03-23 18:28:02

nlinks=1

tamEnBytesLog=209008

numBloquesOcupados=7

**\$ ./leer\_sf disco**



### DATOS DEL SUPERBLOQUE

```
...
posPrimerInodoLibre = 2
cantBloquesLibres = 96854
cantInodosLibres = 24998
...
```

### \$ ./truncar disco 1 9005

```
[liberar_bloques_inodo()→ primer BL: 9, último BL: 204]
[liberar_bloques_inodo()→ liberado BF 3140 de datos para BL 9]
[liberar_bloques_inodo()→ liberado BF 3141 de datos para BL 10]
[liberar_bloques_inodo()→ liberado BF 3142 de datos para BL 11]
[liberar_bloques_inodo()→ liberado BF 3144 de datos para BL 12]
[liberar_bloques_inodo()→ liberado BF 3145 de datos para BL 204]
[liberar_bloques_inodo()→ liberado BF 3143 de punteros_nivel1 correspondiente al BL 204]
[liberar_bloques_inodo()→ total bloques liberados: 6]
liberados: 6
```

### DATOS INODO 1:

```
tipo=f
permisos=6
atime: Tue 2021-03-23 18:28:02
ctime: Tue 2021-03-23 18:28:02
mtime: Tue 2021-03-23 18:28:02
nlinks=1
tamEnBytesLog=9005
numBloquesOcupados=1
```

### \$ ./leer\_sf disco

### DATOS DEL SUPERBLOQUE

```
posPrimerBloqueMB = 1
posUltimoBloqueMB = 13
posPrimerBloqueAI = 14
posUltimoBloqueAI = 3138
posPrimerBloqueDatos = 3139
posUltimoBloqueDatos = 99999
posInodoRaiz = 0
posPrimerInodoLibre = 2
cantBloquesLibres = 96860
cantInodosLibres = 24998
totBloques = 100000
totInodos = 25000
```

### \$ ./truncar disco 1 0

```
[liberar_bloques_inodo()→ primer BL: 0, último BL: 8]
[liberar_bloques_inodo()→ liberado BF 3139 de datos para BL 8]
[liberar_bloques_inodo()→ total bloques liberados: 1]
```

DATOS INODO 1:

tipo=l

permisos=6

atime: Tue 2021-03-23 18:28:02

ctime: Tue 2021-03-23 18:28:02

mtime: Tue 2021-03-23 18:28:02

nlinks=1

tamEnBytesLog=0

numBloquesOcupados=0

**\$ ./leer\_sf disco**

DATOS DEL SUPERBLOQUE

...

posPrimerInodoLibre = 1

cantBloquesLibres = 96861

cantInodosLibres = 24999

totBloques = 100000

totInodos = 25000