Nivel 9: **directorios.c** {mi_write(), mi_read()}, **y mi_escribir.c, mi_cat.c**

comandos:	directorios.c:	ficheros.c:
mi_mkdir, [mi_touch]	mi_creat()	
mi_ls	mi_dir()	
mi_chmod	mi_chmod()	mi_chmod_f()
mi_stat	mi_stat()	mi_stat_f()
mi_escribir	**mi_write()**	**mi_write_f()**
mi_cat	**mi_read()**	**mi_read_f()**
mi_link	mi_link()	
mi_rm, [mi_rmdir], [mi_rm_r]	mi_unlink()	mi_truncar_f()

Tabla con la correlación de comandos y funciones de la capa de directorios y la capa de ficheros

5)Escritura en un offset de un fichero

5a) **mi_escribir.c**

Sintaxis: `./mi_escribir <disco> </ruta_fichero> <texto> <offset>`

Permite escribir texto en una posición de un fichero (*offset*). Podéis adaptar **escribir.c** de la entrega parcial (incluyendo ahora **directorios.h** en vez de **ficheros.h**):

- Ha de recibir como parámetro la **ruta del fichero** en vez del **nº de inodo** (y ya no hay que llamar a reservar_inodo() de forma artificial ya que eso lo hará de forma natural buscar_entrada()).
- Eliminar el argumento de diferentes_inodos y añadir el de offset para poder indicar **desde consola** dónde vamos a escribir[1].
- En vez de llamar a mi_write_f() ha de llamar a mi_write().

[1] Por tanto ya no hay que escribir en diferentes offsets prefijados sino solo en el que nos indiquen desde consola.

Hay que comprobar que se trate de un **fichero**, y tendrá que tener **permisos de escritura** (de esto último se encargará mi_write_f() de la capa de ficheros, que será llamada por mi_write() de la capa de directorios).

Mostrar la cantidad de **bytes escritos**.

5b) int **mi_write**(const char **camino*, const void **buf*, unsigned int *offset*, unsigned int *nbytes*);

Función de **directorios.c** para escribir contenido en un fichero. Buscaremos la entrada camino con buscar_entrada() para obtener el p_inodo. Si la entrada existe llamamos a la función correspondiente de **ficheros.c** pasándole el p_inodo:

```
mi_write_f(p_inodo, buf, offset, nbytes);
```

Ha de devolver los **bytes escritos**.

Observaciones:

- Se puede **optimizar** la operación de escritura utilizando una **caché de directorios**: guardando el camino de la **última entrada buscada** y su p_inodo correspondiente (o un array de las **n últimas entradas**), dado que la mayoría de las lecturas/escrituras seguidas se suelen hacer sobre el mismo inodo ("**principio de proximidad**"), de esta forma, garantizaríamos que solo se busque la entrada si no coincide con la/s última/s buscada/s.
  - Definiríamos, en **directorios.h**, un *struct* con el **camino** asociado a un **nº de inodo**:

```
struct UltimaEntrada{
  char camino [TAMNOMBRE*PROFUNDIDAD];
  int p_inodo;
};
```

donde

```
#define TAMNOMBRE 60 //tamaño del nombre de directorio o fichero
#define PROFUNDIDAD 32 //profundidad máxima del árbol de directorios
```

- Y en **directorios.c** una variable global:

```
static struct UltimaEntrada UltimaEntradaEscritura;
```

- Comprobaríamos si la escritura es sobre el mismo inodo:

```
strcmp(UltimaEntradaEscritura.camino, camino) == 0
```

- Si lo es, entonces:

```
p_inodo = UltimaEntradaEscritura.p_inodo;
```

si no llamar a buscar_entrada() y actualizar los campos de UltimaEntradaEscritura con el p_inodo obtenido para el camino buscado.

También podéis utilizar un array de registros tipo UltimaEntrada, en el que la posición 0 sea por ejemplo para escritura y la posición 1 para lectura.

O incluso tener una caché de más profundidad utilizando un **array de n últimas entradas** de lectura o de escritura (por ejemplo n=5). Para la gestión de esta tabla podéis probar diferentes estrategias: FIFO, LRU, … [2].

En el caso de FIFO, para no desplazar todos los elementos cuando se inserte uno nuevo estando la tabla llena, se puede guardar la posición de la última inserción y tratar la tabla de manera circular mediante el operador módulo.

En el caso de LRU se podría grabar, para cada entrada de la tabla, un sello de tiempo en microsegundos usando la función gettimeofday() y el struct timeval.

6)Lectura secuencial de todo el contenido de un fichero

6a) **mi_cat.c**

Sintaxis: `./mi_cat <disco> </ruta_fichero>`

Programa (comando) que muestra **TODO** el contenido de un fichero. Podéis adaptar **leer.c** de la entrega parcial para que reciba como parámetro la **ruta del fichero** en vez del **nº de inodo** (incluyendo ahora **directorios.h** en vez de **ficheros.h**) y ha de llamar a mi_read() en vez de a mi_read_f().

Observaciones:

- **Hay que comprobar que la ruta se corresponda a un fichero** ya que si es un directorio no podemos hacer un cat.
- Tendrán que coincidir los bytes leídos con el tamaño en bytes lógico del fichero y con el tamaño físico del fichero externo al que **redireccionemos** la lectura, y se ha de **filtrar** la basura.
- Utilizar una variable por ej. tambuffer para poder cambiar el **tamaño del buffer de lectura** sin tener que modificar todas las sentencias involucradas. Recordad probar con diferentes tamaños, tanto múltiplos de varios bloques como con un valor cualquiera, como por ejemplo 1500 bytes.

[2] Ver algoritmos de reemplazo de paginación para memoria principal.

6b) int **mi_read**(const char *camino, void *buf, unsigned int offset, unsigned int nbytes);

Función de **directorios.c** para leer los nbytes del fichero indicado por camino, a partir del offset pasado por parámetro y copiarlos en el buffer buf. Buscaremos la entrada camino con buscar_entrada() para obtener el p_inodo. Si la entrada existe llamamos a la función correspondiente de **ficheros.c** pasándole el p_inodo:

```
mi_read_f(p_inodo, buf, offset, nbytes);
```

Ha de devolver los **bytes leidos**.

También se puede **optimizar** utilizando una **caché de directorios** guardando el camino y p_inodo de la **última lectura** o de las **n últimas lecturas**.

TESTS DE PRUEBA [3]

Podéis ejecutar linea a linea[4] o usar el script **test9.sh** (lo adaptáis si es preciso, éste utiliza **mi_touch** y **comprueba la caché de directorios que guarda 1 entrada para escritura y otra para lectura**). Para probar una caché tipo tabla FIFO o LRU hay que crear otro script.

```
uib:~$ ./test9.sh

##############################################################
$ ./mi_mkfs disco 100000

##############################################################
$ ./mi_touch disco 6 /fichero

$ ./mi_ls disco /
Total: 1
Tipo    Modo  mTime                    Tamaño              Nombre
--------------------------------------------------------------------------------
f       rw-   2022-06-02 14:20:12      0                   fichero

$ ./mi_escribir  disco /fichero hola que tal 5120
longitud texto: 12
[mi_write() → Actualizamos la caché de escritura]
Bytes escritos: 12
```

[3] Aquí ya se pueden omitir los fprintf() de buscar_entrada() salvo los errores y dejar comentadas las llamadas a buscar_entrada() de **leer_sf.c**
[4] Ejecutado línea a línea manualmente observaréis que varían los sellos de tiempo entre sí.

$ ./mi_ls disco /
Total: 1
Tipo Modo mTime Tamaño Nombre
---
f rw- 2022-06-02 14:20:12 5132 fichero

$ ./leer_sf disco

DATOS DEL SUPERBLOQUE
posPrimerBloqueMB = 1
posUltimoBloqueMB = 13
posPrimerBloqueAI = 14
posUltimoBloqueAI = 3138
posPrimerBloqueDatos = 3139
posUltimoBloqueDatos = 99999
posInodoRaiz = 0
posPrimerInodoLibre = 2
cantBloquesLibres = 96859
cantInodosLibres = 24998
totBloques = 100000
totInodos = 25000

##
$ ./mi_chmod disco 4 /fichero

$ ./mi_escribir disco /fichero estoy estupendamente 256000
longitud texto: 20
[mi_write() → Actualizamos la caché de escritura]
No hay permisos de escritura
Bytes escritos: 0

$ ./mi_ls disco /fichero #si no habéis implementado mi_ls para ficheros toca dar error
Tipo Modo mTime Tamaño Nombre
---
f r-- 2022-06-02 14:20:12 5132 fichero

##
$ ./mi_mkdir disco 6 /dir1/

$ ./mi_touch disco 6 /dir1/fic1

./mi_escribir disco /dir1/fic1 hola1 256000
longitud texto: 5
[mi_write() → Actualizamos la caché de escritura]
Bytes escritos: 5

$ ./mi_stat disco /dir1/fic1
Nº de inodo: 3
tipo: f
permisos: 6
atime: Thu 2022-06-02 14:20:12
ctime: Thu 2022-06-02 14:20:12
mtime: Thu 2022-06-02 14:20:12
nlinks: 1
tamEnBytesLog: 256005
numBloquesOcupados: 2

##
$ sleep 1 #esperamos un poco para observar los sellos de tiempo

##
$ ./mi_escribir disco /dir1/fic1 hola2 5120 #no cambia tamenBytesLog pero sí mtime y ctime (ocupamos 1 bloque más)
longitud texto: 5
[mi_write() → Actualizamos la caché de escritura]
Bytes escritos: 5

$ ./mi_stat disco /dir1/fic1
Nº de inodo: 3
tipo: f
permisos: 6
atime: Thu 2022-06-02 14:20:12
ctime: Thu 2022-06-02 14:20:13
mtime: Thu 2022-06-02 14:20:13
nlinks: 1
tamEnBytesLog: 256005
numBloquesOcupados: 3

##
$ sleep 1 #esperamos un poco para observar los sellos de tiempo

##
$ ./mi_escribir disco /dir1/fic1 hola3 5200 #mismo bloque que offset 5120, cambia mtime pero no ctime
longitud texto: 5
[mi_write() → Actualizamos la caché de escritura]
Bytes escritos: 5

$ ./mi_stat disco /dir1/fic1
Nº de inodo: 3

tipo: f
permisos: 6
atime: Thu 2022-06-02 14:20:12
ctime: Thu 2022-06-02 14:20:13
mtime: Thu 2022-06-02 14:20:15
nlinks: 1
tamEnBytesLog: 256005
numBloquesOcupados: 3

##
sleep 1 #esperamos un poco para observar los sellos de tiempo

##
./mi_escribir disco /dir1/fic1 hola4 256010 #cambia tamEnBytesLog, mtime y ctime
longitud texto: 5
[mi_write() → Actualizamos la caché de escritura]
Bytes escritos: 5

$ ./mi_stat disco /dir1/fic1
Nº de inodo: 3
tipo: f
permisos: 6
atime: Thu 2022-06-02 14:20:12
ctime: Thu 2022-06-02 14:20:16
mtime: Thu 2022-06-02 14:20:16
nlinks: 1
tamEnBytesLog: 256015
numBloquesOcupados: 3

##
 # Comprobación de la caché de directorios (1 entrada para L y otra para E)
##
$ ./mi_touch disco 6 /dir1/fic2

$ ./mi_escribir disco /dir1/fic2 ¿Qué es Lorem Ipsum? Lorem Ipsum es simplemente el texto de relleno de las imprentas y archivos de texto. Lorem Ipsum ha sido el texto de ...
repeticiones, humor agregado o palabras no características del lenguaje, etc. 1000
longitud texto: 3751
[mi_write() → Actualizamos la caché de escritura]
Bytes escritos: 3751
##
$ ./mi_cat disco /dir1/fic2 #tambuffer=BLOCKSIZE * 4
[mi_read() → Actualizamos la caché de lectura]

¿Qué es Lorem Ipsum?

Lorem Ipsum es simplemente el texto de relleno de las imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno estándar de las industrias desde el año 1500, cuando un impresor (N. del T. persona que se dedica a la imprenta) desconocido usó una galería de textos y los mezcló de tal manera que logró hacer un libro de textos especimen. No sólo sobrevivió 500 años, sino que tambien ingresó como texto de relleno en documentos electrónicos, quedando esencialmente igual al original. Fue popularizado en los 60s con la creación de las hojas "Letraset", las cuales contenian pasajes de Lorem Ipsum, y más recientemente con software de autoedición, como por ejemplo Aldus PageMaker, el cual incluye versiones de Lorem Ipsum.
¿Por qué lo usamos?

Es un hecho establecido hace demasiado tiempo que un lector se distraerá con el contenido del texto de un sitio mientras que mira su diseño. El punto de usar Lorem Ipsum es que tiene una distribución más o menos normal de las letras, al contrario de usar textos como por ejemplo "Contenido aquí, contenido aquí". Estos textos hacen parecerlo un español que se puede leer. Muchos paquetes de autoedición y editores de páginas web usan el Lorem Ipsum como su texto por defecto, y al hacer una búsqueda de "Lorem Ipsum" va a dar por resultado muchos sitios web que usan este texto si se encuentran en estado de desarrollo. Muchas versiones han evolucionado a través de los años, algunas veces por accidente, otras veces a propósito (por ejemplo insertándole humor y cosas por el estilo).

¿De dónde viene?

Al contrario del pensamiento popular, el texto de Lorem Ipsum no es simplemente texto aleatorio. Tiene sus raices en una pieza cl´sica de la literatura del Latin, que data del año 45 antes de Cristo, haciendo que este adquiera mas de 2000 años de antiguedad. Richard McClintock, un profesor de Latin de la Universidad de Hampden-Sydney en Virginia, encontró una de las palabras más oscuras de la lengua del latín, "consecteur", en un pasaje de Lorem Ipsum, y al seguir leyendo distintos textos del latín, descubrió la fuente indudable. Lorem Ipsum viene de las secciones 1.10.32 y 1.10.33 de "de Finnibus Bonorum et Malorum" (Los Extremos del Bien y El Mal) por Cicero, escrito en el año 45 antes de Cristo. Este libro es un tratado de teoría de éticas, muy popular durante el Renacimiento. La primera linea del Lorem Ipsum, "Lorem ipsum dolor sit amet..", viene de una linea en la sección 1.10.32

El trozo de texto estándar de Lorem Ipsum usado desde el año 1500 es reproducido debajo para aquellos interesados. Las secciones 1.10.32 y 1.10.33 de "de Finibus Bonorum et Malorum" por Cicero son también reproducidas en su forma original exacta, acompañadas por versiones en Inglés de la traducción realizada en 1914 por H. Rackham.
¿Dónde puedo conseguirlo?

Hay muchas variaciones de los pasajes de Lorem Ipsum disponibles, pero la mayoría sufrió alteraciones en alguna manera, ya sea porque se le agregó humor, o palabras aleatorias que no parecen ni un
[mi_read() → Utilizamos la caché de lectura en vez de llamar a buscar_entrada()]
poco creíbles. Si vas a utilizar un pasaje de Lorem Ipsum, necesitás estar seguro de que no hay nada avergonzante escondido en el medio del texto. Todos los generadores de Lorem Ipsum que se encuentran en Internet tienden a repetir trozos predefinidos cuando sea necesario, haciendo a este el único generador verdadero (válido) en la Internet. Usa un diccionario de mas de 200 palabras provenientes del latín, combinadas con estructuras muy útiles de sentencias, para generar texto de Lorem Ipsum que parezca razonable. Este Lorem Ipsum generado siempre estará libre de repeticiones, humor agregado o palabras no características del lenguaje, etc.
[mi_read() → Utilizamos la caché de lectura en vez de llamar a buscar_entrada()]

Total_leidos 4751

##
$ ./mi_escribir disco /dir1/fic2 "********************************" 10000**
longitud texto: 30
[mi_write() → Actualizamos la caché de escritura]
Bytes escritos: 30

$ ./mi_ls disco /dir1/
Total: 2

Tipo	Modo	mTime	Tamaño	Nombre
f	rw-	2022-06-02 14:20:16	256015	fic1
f	rw-	2022-06-02 14:20:16	10030	fic2

$ ./mi_cat disco /dir1/fic2 #tambuffer=BLOCKSIZE * 4
[mi_read() → Actualizamos la caché de lectura]

¿Qué es Lorem Ipsum?

Lorem Ipsum es simplemente el texto de relleno de las imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno estándar de las industrias desde el año 1500, cuando un impresor (N. del T. persona que se dedica a la imprenta) desconocido usó una galería de textos y los mezcló de tal manera que logró hacer un libro de textos especimen. No sólo sobrevivió 500 años, sino que tambien ingresó como texto de relleno en documentos electrónicos, quedando esencialmente igual al original. Fue popularizado en los 60s con la creación de las hojas "Letraset", las cuales contenian pasajes de Lorem Ipsum, y más recientemente con software de autoedición, como por ejemplo Aldus PageMaker, el cual incluye versiones de Lorem Ipsum.
¿Por qué lo usamos?

Es un hecho establecido hace demasiado tiempo que un lector se distraerá con el contenido del texto de un sitio mientras que mira su diseño. El punto de usar Lorem Ipsum es que tiene una distribución más o menos normal de las letras, al contrario de usar textos como por ejemplo "Contenido aquí, contenido aquí". Estos textos hacen parecerlo un español que se puede leer. Muchos paquetes de autoedición y editores de páginas web usan el Lorem Ipsum como su texto por defecto, y al hacer una búsqueda de "Lorem Ipsum" va a dar por resultado muchos sitios web que usan este texto si se encuentran en estado de desarrollo. Muchas versiones han evolucionado a través de los años, algunas veces por accidente, otras veces a propósito (por ejemplo insertándole humor y cosas por el estilo).

¿De dónde viene?

Al contrario del pensamiento popular, el texto de Lorem Ipsum no es simplemente texto aleatorio. Tiene sus raices en una pieza cl´sica de la literatura del Latin, que data del año 45 antes de Cristo, haciendo que este adquiera mas de 2000 años de antiguedad. Richard McClintock, un profesor de Latin de la Universidad de Hampden-Sydney en Virginia, encontró una de las palabras más oscuras de la lengua del latín, "consecteur", en un pasaje de Lorem Ipsum, y al seguir leyendo distintos textos del latín, descubrió la fuente indudable. Lorem Ipsum viene de las secciones 1.10.32 y 1.10.33 de "de Finnibus Bonorum et Malorum" (Los Extremos del Bien y El Mal) por Cicero, escrito en el año 45 antes de Cristo. Este libro es un tratado de teoría de éticas, muy popular durante el Renacimiento. La primera linea del Lorem Ipsum, "Lorem ipsum dolor sit amet..", viene de una linea en la sección 1.10.32

El trozo de texto estándar de Lorem Ipsum usado desde el año 1500 es reproducido debajo para aquellos interesados. Las secciones 1.10.32 y 1.10.33 de "de Finibus Bonorum et Malorum" por Cicero son también reproducidas en su forma original exacta, acompañadas por versiones en Inglés de la traducción realizada en 1914 por H. Rackham.
¿Dónde puedo conseguirlo?

Hay muchas variaciones de los pasajes de Lorem Ipsum disponibles, pero la mayoría sufrió alteraciones en alguna manera, ya sea porque se le agregó humor, o palabras aleatorias que no parecen ni un
[mi_read() → Utilizamos la caché de lectura en vez de llamar a buscar_entrada()]
poco creíbles. Si vas a utilizar un pasaje de Lorem Ipsum, necesitás estar seguro de que no hay nada avergonzante escondido en el medio del texto. Todos los generadores de Lorem Ipsum que se encuentran en Internet tienden a repetir trozos predefinidos cuando sea necesario, haciendo a este el único generador verdadero (válido) en la Internet. Usa un diccionario de mas de 200 palabras provenientes del latín, combinadas con estructuras muy útiles de sentencias, para generar texto de Lorem Ipsum que parezca razonable. Este Lorem Ipsum generado siempre estará libre de repeticiones, humor agregado o palabras no características del lenguaje, etc.

```
[mi_read() → Utilizamos la caché de lectura en vez de llamar a buscar_entrada()]
******************************
[mi_read() → Utilizamos la caché de lectura en vez de llamar a buscar_entrada()]


Total_leidos 10030
#################################################################
$ ./mi_touch disco 6 /dir1/fic3

$ ./mi_escribir_varios disco /dir1/fic3 "--texto repetido en 10 bloques--" 0
longitud texto: 32
[mi_write() → Actualizamos la caché de escritura]
[mi_write() → Utilizamos la caché de escritura en vez de llamar a buscar_entrada()]
[mi_write() → Utilizamos la caché de escritura en vez de llamar a buscar_entrada()]
[mi_write() → Utilizamos la caché de escritura en vez de llamar a buscar_entrada()]
[mi_write() → Utilizamos la caché de escritura en vez de llamar a buscar_entrada()]
[mi_write() → Utilizamos la caché de escritura en vez de llamar a buscar_entrada()]
[mi_write() → Utilizamos la caché de escritura en vez de llamar a buscar_entrada()]
[mi_write() → Utilizamos la caché de escritura en vez de llamar a buscar_entrada()]
[mi_write() → Utilizamos la caché de escritura en vez de llamar a buscar_entrada()]
[mi_write() → Utilizamos la caché de escritura en vez de llamar a buscar_entrada()]
Bytes escritos: 320

$ ./mi_stat disco /dir1/fic3
Nº de inodo: 5
tipo: f
permisos: 6
atime: Thu 2022-06-02 14:20:16
ctime: Thu 2022-06-02 14:20:16
mtime: Thu 2022-06-02 14:20:16
nlinks: 1
tamEnBytesLog: 9248
numBloquesOcupados: 10

$ ./mi_cat disco /dir1/fic3 #tambuffer=BLOCKSIZE * 4
[mi_read() → Actualizamos la caché de lectura]
--texto repetido en 10 bloques----texto repetido en 10 bloques----texto repetido en 10
bloques----texto repetido en 10 bloques--
[mi_read() → Utilizamos la caché de lectura en vez de llamar a buscar_entrada()]
--texto repetido en 10 bloques----texto repetido en 10 bloques----texto repetido en 10
bloques----texto repetido en 10 bloques--
[mi_read() → Utilizamos la caché de lectura en vez de llamar a buscar_entrada()]
--texto repetido en 10 bloques----texto repetido en 10 bloques--
[mi_read() → Utilizamos la caché de lectura en vez de llamar a buscar_entrada()]
```

Total_leidos 9248

ANEXO

```
//mi_escribir_varios.c
//Programa para testear las cachés de directorios

#include "directorios.h"

int main(int argc, char **argv){

 //Comprobamos sintaxis
 if (argc!=5) {
   fprintf(stderr, "Sintaxis: mi_escribir_varios <nombre_dispositivo> </ruta_fichero> <texto>
<offset>\n");
   exit(-1);
  }

 //struct STAT stat;

 //montamos el dispositivo
 if(bmount(argv[1])<0) return -1;
 //obtenemos el texto y su longitud
 char *buffer_texto = argv[3];
 int longitud=strlen(buffer_texto);

 //obtenemos la ruta y comprobamos que no se refiera a un directorio
 if (argv[2][strlen(argv[2])-1]=='/') {
   fprintf(stderr, "Error: la ruta se corresponde a un directorio");
   exit(-1);
 }
 char *camino = argv[2];
 //obtenemos el offset
 unsigned int offset=atoi(argv[4]);
 int escritos=0;
 int varios = 10;
 fprintf(stderr, "longitud texto: %d\n", longitud);
 for (int i=0; i<varios; i++) {
   // escribimos varias veces el texto desplazado 1 bloque
   escritos += mi_write(camino,buffer_texto,offset+BLOCKSIZE*i,longitud);
 }
 fprintf(stderr, "Bytes escritos: %d\n", escritos);
 /* Visualización del stat
```

```
mi_stat_f(ninodo, &stat);
printf("stat.tamEnBytesLog=%d\n",stat.tamEnBytesLog);
printf("stat.numBloquesOcupados=%d\n",stat.numBloquesOcupados);
*/

bumount();
}
```

Programa para poder testear una **caché de directorios de tamaño > 1**. Genera 3 directorios y 3 subdirectorios dentro de cada uno con un fichero y luego genera escrituras seleccionando aleatoriamente una de esas rutas. **Tenéis que mostrar cuando y a qué elemento de la tabla caché se está accediendo**.

```
//Programa mi_escribir_varios_difdirs.c para testear las cachés de directorios > 1
//Sintaxis: mi_escribir_varios_difdirs <nombre_dispositivo> <texto>
#include "directorios.h"
#define DIFDIRS 3
//Se crearán tantos directorios como indique ese valor y dentro de cada uno también

int main(int argc, char **argv){

 char camino[256];
 memset(camino, 0 , sizeof(camino));
 unsigned int offset=0;
 unsigned int escritos=0;

 //Comprobamos sintaxis
 if (argc!=3) {
  fprintf(stderr, "Sintaxis: mi_escribir_varios_difdirs <nombre_dispositivo> <texto> \n");
  exit(-1);
 }

 //montamos el dispositivo
 if(bmount(argv[1])<0) return -1;
 //obtenemos el texto y su longitud
 char *buffer_texto = argv[2];
 int longitud=strlen(buffer_texto);

 //creación de subdirectorios y ficheros
 for (int i=0; i<DIFDIRS; i++){
    sprintf (camino, "/dir%d/", i);
    if (mi_creat (camino, 6) < 0) {
       fprintf (stderr, "Error al crear directorio %s\n", camino);
       bumount();
       exit(EXIT_FAILURE);
```

```
    } else
      fprintf(stderr,"[mi_escribir_varios_difdirs.c → Creado directorio %s]\n", camino);
    for (int j=0; j<DIFDIRS; j++){
      sprintf (camino, "/dir%d/dir%d_%d/", i, i, j);
      if (mi_creat (camino, 6) < 0) {
        fprintf (stderr, "Error al crear directorio %s\n", camino);
        bumount();
        exit(EXIT_FAILURE);
      } else fprintf(stderr,"[mi_escribir_varios_difdirs.c → Creado directorio %s]\n", camino);

      strcat(camino, "fichero.txt");
      if (mi_creat (camino, 6) < 0) {
        fprintf (stderr, "Error al crear fichero %s\n", camino);
        bumount();
        exit(EXIT_FAILURE);
      } else fprintf(stderr,"[mi_escribir_varios_difdirs.c → Creado directorio %s]\n", camino);

      memset(camino,0,sizeof(camino));
    }
    memset(camino,0,sizeof(camino));
  }

  //escrituras
  srand(time(NULL));
  for (int i=0; i<DIFDIRS*100; i++){
    int num1 = rand()%DIFDIRS;
    int num2 = rand()%DIFDIRS;
    sprintf (camino, "/dir%d/dir%d_%d/fichero.txt", num1, num1, num2);
    escritos += mi_write(camino,buffer_texto,offset,longitud);
    fprintf(stderr,"[mi_escribir_varios_difdirs.c → Bytes escritos en %s: %d]\n",  camino,
escritos);
    escritos = 0;
  }
  bumount();
}
```