Introducción al entorno de desarrollo

Sistemas Operativos
Práctica 1

Práctica 1: Programa “mitar”

- El guión de la práctica puede encontrarse en el CV

Archivo mtar: fichero binario que alberga múltiples ficheros en su interior

Implementación

- Modo de uso

```
$ ./mytar -c|x -f archivo_mtar [fich1 fich2 …]
 •  -c : Crear archivo mtar
      ○  Ejemplo: $ ./mytar -c -f ejemplo.mtar a.txt b.txt
 •  -x : Extraer archivo mtar
      ○  Ejemplo: $ ./mytar -x -f ejemplo.mtar
```

- El proyecto consta de los siguientes ficheros
  - `Makefile`
  - `mytar.c` : función main() del programa
  - `mytar.h` : declaraciones de tipos de datos y funciones
  - `mytar_routines.c` : funciones de creación y extracción de ficheros mtar
    - Único fichero a modificar

Implementación

```
// mytar.h

#ifndef _MYTAR_H
#define _MYTAR_H

#include <limits.h>

typedef enum{
  NONE,
  ERROR,
  CREATE,
  EXTRACT
} flags;

typedef struct {
  char *name;
  unsigned int size;
} stHeaderEntry;

int createTar(int nFiles, char *fileNames[], char tarName[]);
int extractTar(char tarName[]);

#endif /* _MYTAR_H */
```

Funciones a implementar

```
int createTar(int nFiles, char *fileNames[], char tarName[]);
/* Creates a tarball archive */

int extractTar(char tarName[]);
/* Extract files stored in a tarball archive */

int copynFile(FILE *origen, FILE *destimo, int nBytes);
/* Copy nBytes bytes from the origin file to the destination file. */

int loadstr(FILE *file);
/* Loads a string from a file. */

int readHeader(FILE *tarFile, int *nFiles);
/* Read tarball header and store it in memory. */
```

Extracción del fichero mtar

- En la extracción del fichero mtar tenemos un problema
  - No sabemos cuánto ocupa cada nombre de fichero, tenemos que leer hasta encontrar '\0'

Estructura de readHeader

```
stHeaderEntry* readHeader(FILE *tarFile, int *nFiles)
{
     int i,j;
     stHeaderEntry* header;

     ... Leemos el número de ficheros (N) del tarFile y lo copiamos en nFiles

     /* Reservamos memoria para el array */
     header = (stHeaderEntry *) malloc(sizeof (stHeaderEntry) * (*nFiles));

     for (i = 0; i < *nFiles; i++) {
          ... usamos loadstr para cargar el nombre en header[i].name
          ... comprobación y tratamiento de errores
          ... leemos el tamaño del fichero y lo almacenamos en header[i].size
     }

     return header;
}
```

Creación del fichero mtar

- En la creación del fichero mtar tenemos algunos problemas
  - No sabemos de antemano lo que va a ocupar la cabecera, depende de la suma de los tamaños de los nombre/rutas de los ficheros a introducir en el tar
  - No sabemos de antemano cuál es el tamaño en bytes de cada uno de los ficheros que hay que introducir en el mtar

createTar: implementación simple ¿Se puede hacer más eficiente?

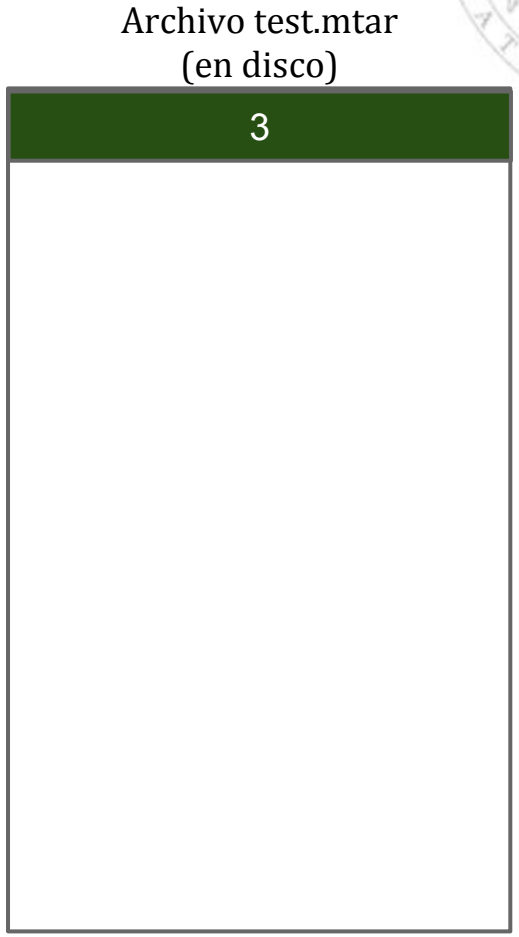
1. Abrimos el fichero mtar para escritura (fichero destino)
2. Reservamos memoria (con `malloc()`) para un array de `stHeaderEntry`
   - El array tendrá tantas posiciones como ficheros haya que introducir en el mtar
3. Copiamos la cabecera al fichero mtar
   - Copiamos nFiles
   - Por cada fichero (fileNames):
     - Copiamos el nombre terminado en '\0' al mtar
     - Calculamos el tamaño del fichero (fopen, fseek, fclose)
     - Copiamos el tamaño al mtar
4. Por cada fichero (fileNames):
   - Abrimos el fichero
   - Lo copiamos a disco con copynFile
   - Cerramos el fichero
5. Cerramos el fichero mtar

Ejemplo: Creación de un fichero mtar

```
$ ./mitar -c -f test.mtar a.txt b.txt c.txt
```

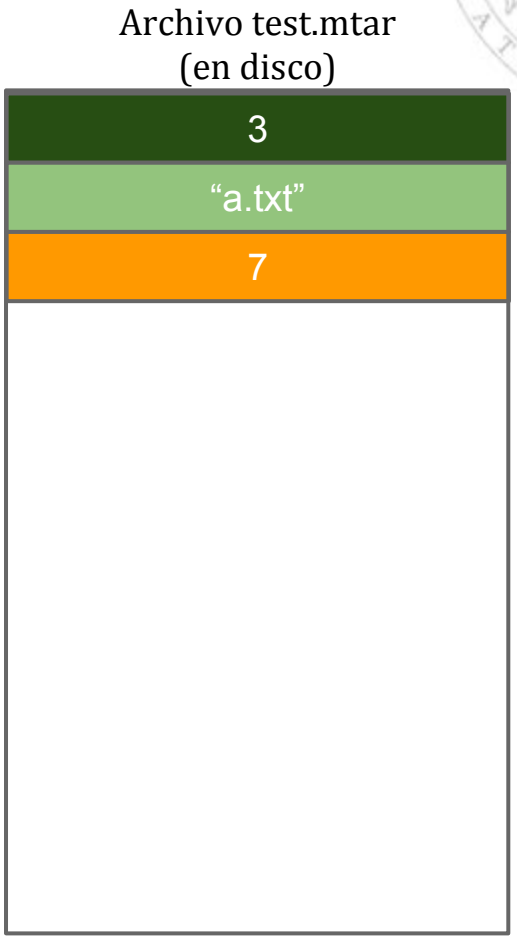
Ejemplo: Creación de un fichero mtar

```
$ ./mitar -c -f test.mtar a.txt b.txt c.txt
```



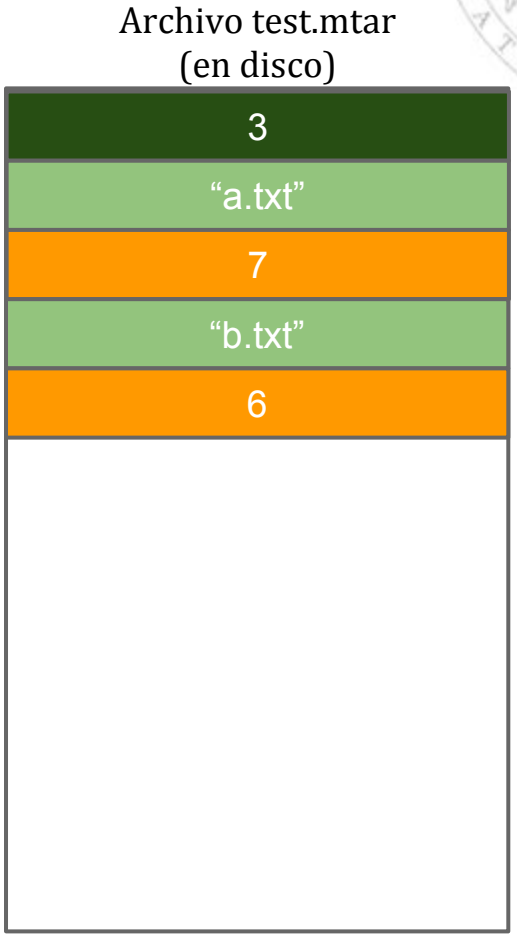
Ejemplo: Creación de un fichero mtar

```
$ ./mitar -c -f test.mtar a.txt b.txt c.txt
```



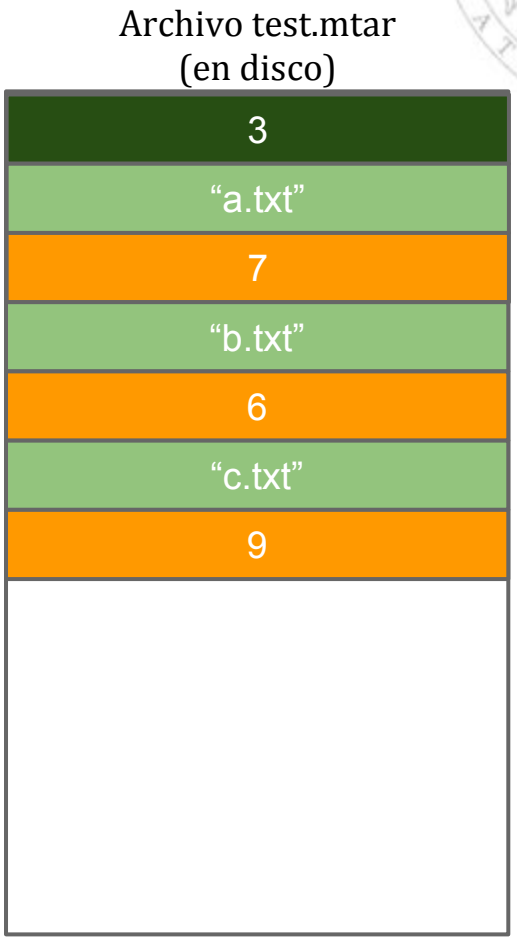
Ejemplo: Creación de un fichero mtar

```
$ ./mitar -c -f test.mtar a.txt b.txt c.txt
```



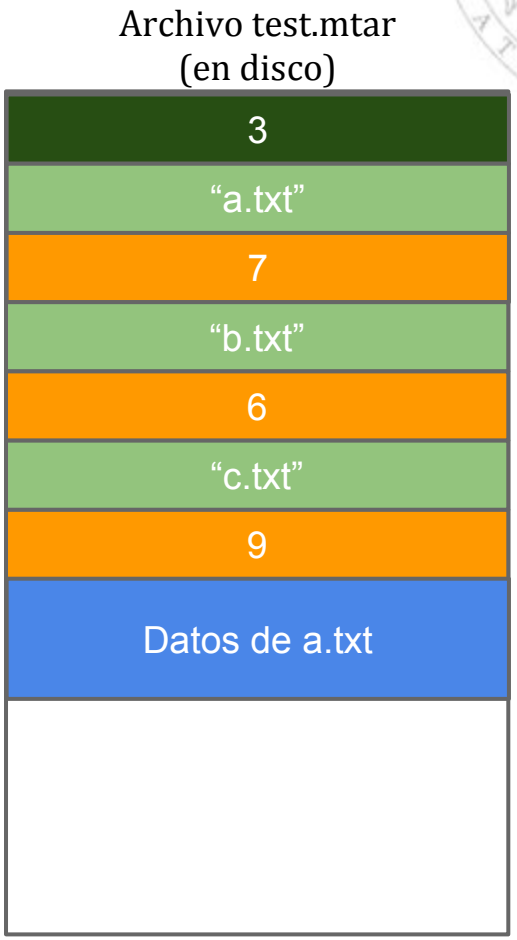
Ejemplo: Creación de un fichero mtar

```
$ ./mitar -c -f test.mtar a.txt b.txt c.txt
```



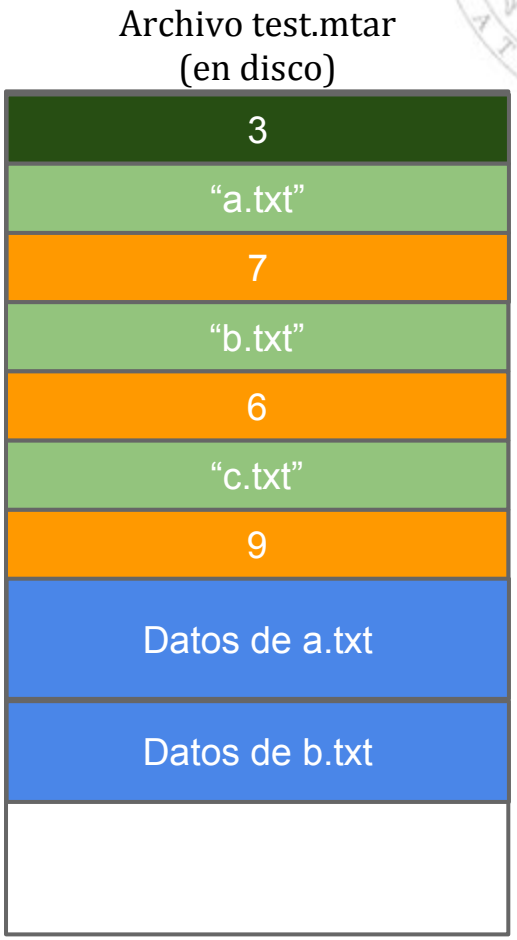
Ejemplo: Creación de un fichero mtar

```
$ ./mitar -c -f test.mtar a.txt b.txt c.txt
```



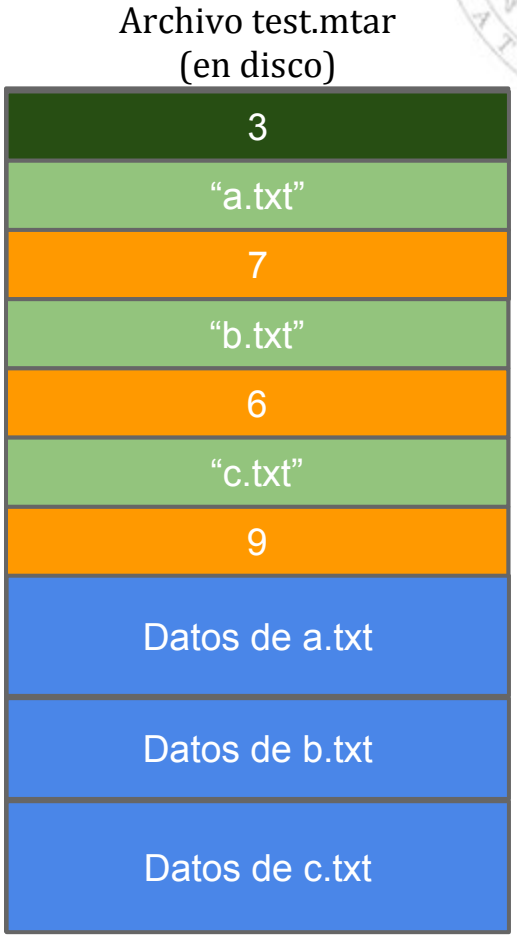
Ejemplo: Creación de un fichero mtar

```
$ ./mitar -c -f test.mtar a.txt b.txt c.txt
```



Ejemplo: Creación de un fichero mtar

```
$ ./mitar -c -f test.mtar a.txt b.txt c.txt
```



Ejemplo de ejecución

```
$ ls
a.txt  b.txt  c.txt  makefile  mitar.c  mitar.h  rut_mitar.c
$ du -b *.txt
7    a.txt
6    b.txt
9    c.txt
$ make
gcc -g -Wall  -c  mitar.c -o mitar.o
gcc -g -Wall  -c  rut_mitar.c -o rut_mitar.o
gcc -g -Wall  -o mitar mitar.o rut_mitar.o
$ ./mytar -c -f test.mtar a.txt b.txt c.txt
Fichero mitar creado con éxito
$ ls
a.txt  c.txt     mitar    mitar.h  rut_mitar.c  test.mtar
b.txt  makefile  mitar.c  mitar.o  rut_mitar.o
```

Ejemplo de ejecución

```
$ mkdir tmp
$ cd tmp/
$ ../mytar -x -f ../test.mtar
[0]: Creando fichero a.txt, tamano 7 Bytes...Ok
[1]: Creando fichero b.txt, tamano 6 Bytes...Ok
[2]: Creando fichero c.txt, tamano 9 Bytes...Ok
$ ls
a.txt  b.txt  c.txt
$ diff a.txt ../a.txt
$ diff b.txt ../b.txt
$ diff c.txt ../c.txt
```