```c
/*
 * Archivo extract.c
 *
 * Descripcion: Archivo fuente con las funciones necesarias para la extraccion
 *  de un archivo .mytar
 *
 * Autores:
 *  Carlos Alejandro Sivira Munoz     15-11377
 *  Cesar Alfonso Rosario Escobar     15-11295
 */

#ifndef __EXTRACT__
#define __EXTRACT__

#include "parser.h"

#define STUFF_TOKEN '•'
#define CREATION_MODE O_WRONLY | O_TRUNC | O_CREAT
#define max(X,Y) ((X > Y)? X: Y)
#define MAX_RW 1

/* struct f_att
 * ----------
 *  Estructura en la que se almacenan los atributos correspondientes
 *  a la metadata de un archivo. Estos se usan luego para crear al archivo.
 *
 */
typedef struct {
  mode_t mode;
  uid_t uid;
  gid_t gid;
  long size;
  char* name;
  char* link_ptr;
} f_att;

/* fileWriterBounded
 * ----------
 * Escribe de un archivo a otro utilizando los "file descriptors" de ambos.
 * Limita el contenido de escritura a un parametro entero.
 *
 *
 *  fd_source: "file descriptor" del archivo del que se copia.
 *  fd_dest: "file descriptor" del archivo al que se copia.
 *  inst: Estructura que contiene la informacion de las opciones de mytar
 */
void fileWriterBounded(int fd_source, int fd_dest, int total, mytar_instructions
inst) ;


/* getField
 * --------------
 * Devuelve un string que representa un campo de cabecera de archivo .mytar
 *
 *  fd: "file descriptor" del archivo .mytar.
 *  filed_length: tamano del campo de cabecera
 *
 * retorna: "string" correspondiente a un campo de cabecera.
 */
char *getField(int fd, int field_length) ;
```

```
60
61
62 /* getFieldSize
63  * --------------
64  * Calcula el tamano de un campo numerico de cabecera, buscando la siguiente
65  * aparicion del caracter "•" que actua como un separador entre los campos.
66  *
67  *
68  *  fd: "file descriptor" del .mytar
69  *
70  * retorna: Un entero que representa el tamano del campo.
71  */
72 int getFieldSize(int fd) ;
73
74
75 /* putField
76  * --------------
77  *  Esta funcion utiliza la separacion en campos de la cabecera del archivo
78  *  .mytar para adquirir campos numericos y devolverlos como tipo long.
79  *
80  *
81  *    fd: file descriptor del .mytar
82  *
83  *  retorna: un long asociado a un campo numerico de cabecera.
84  */
85 long putField(int fd) ;
86
87
88 /* setModeAndOwn
89  * --------------
90  * Para cualquier archivo, se encarga de modificar su permisos (bits modales)
91  * asi como su dueno y grupo, utilizando ung gid y un uid.
92  *
93  *  attr: Estructura de donde obtiene lo que modifica
94  */
95 void setModeAndOwn(f_att attr);
96
97
98 /* myLs
99  * --------------
100 * Imprime un listado similar al del comando ls -l de los archivos
101 * presentes en el .mytar
102 *
103 *
104 *  attr: Estructura que contiene los atributos del archivo.
105 *  type: Entero que representa el tipo de archivo. 1=regular
106 *    2=directorio, 3=Link simbolico
107 */
108 void myLs(f_att attr, int type ) ;
109
110
111 /* createFile
112  * --------------
113  * Crea un archivo de alguno de los tipos considerados (regulares, directorios,
114  * links simbolicos) y actualiza sus atributos.
115  *
116  *
117  *  fd: "file descriptor" del archivo .mytar
118  *  offset: posicion actual del apuntador en el archivo .mytar
119  *    instructions: Estructura que contiene la informacion de las opciones de
```

```
 *          mytar.
 *
 * Retorna la posicion actual del apuntador
 */
int createFile(int fd, long offset, f_att prueba, mytar_instructions inst) ;


/* gatherFields
 * --------------
 *  Esta funcion junta los campos de cabecera (tanto numericos como no
 *  numericos) correspondientes a la metada de un archivo empaquetado
 *
 *
 *    fd = "file descriptor" del .mytar
 *      instructions: Estructura que contiene la informacion de las opciones de
 *    mytar.
 *
 *  retorna: el offset actual del archivo, o -1 en caso de error.
 */
int gatherFields(int fd, mytar_instructions inst) ;


/* extractMyTar
 * --------------
 * Recibe un archivo .mytar y se encarga de extraer su contenido.
 *
 *  mt_name: Nombre del archivo .mytar a procesar
 *    instructions: Estructura que contiene la informacion de las opciones de
 *          mytar.
 */
int extractMyTar(char** mt_name, mytar_instructions inst);

#endif
```