```c
/*
 * Archivo: create.h
 *
 * descripcion: Archivo de cabecera con las firmas de funciones usadas para
 * crear un archivo .mytar
 *
 * Autores:
 *  Carlos Alejandro Sivira Munoz     15-11377
 *  Cesar Alfonso Rosario Escobar   15-11295
 */

#ifndef __CREATE__
#define __CREATE__

#include "parser.h"

#define MAX_RW 1
#define CREATE_APPEND_MODE O_WRONLY | O_TRUNC | O_CREAT
#define MY_PERM S_IRUSR | S_IWUSR | S_IRGRP | S_IROTH
#define MAX_PATHNAME 5000
#define STUFF_TOKEN '•'

/* setHeadFields
 * ----------
 * Asigna los campos de cabecera del archivo .mytar
 *
 *
 *  fd_dest: "file descriptor" del archivo .mytar
 *  state: Estado del archivo actual.
 *  name: Nombre del archivo a empaquetar.
 */
void setHeadFields(int fd_dest, struct stat state, char *name) ;


/* fileWriter
 * ----------
 * Escribe de un archivo a otro utilizando sus "file descriptors"
 *
 *
 *  fd_source: "file descriptor" del archivo del que se lee
 *  fd_dest: "file descriptor" del archivo al que se escribe
 *  instructions: Estructura que contiene la informacion de las opciones de
 *          mytar.
 */
void fileWriter(int fd_source, int fd_dest, mytar_instructions inst) ;



/* handleFileType
 * --------------
 *  Esta funcion asigna los campos de cabecera .mytar necesarios para
 *  poder guardar la metadata de un archivo.
 *
 *
 *    fd_dest: File descriptor de archivo .mytar.
 *    pathname: nombre del archivo que se esta procesando.
 *    current_st: Estado del archivo.
 *      instructions: Estructura que contiene la informacion de las opciones de
 *      mytar.
 *
```

```
 * Retorna NULL, o DIR*, en caso de que el archivo procesado sea un directorio.
 *
 */
DIR *handleFileType(int fd_dest, char* pathname, struct stat current_st,
mytar_instructions inst) ;


/* traverseDir
 * ----------
 * De haber un arbol de directorios como argumento para archivar, realizando las
llamadas
 * a funciones necesarias para crear el archivo .mytar
 *
 *  dir: apuntador al directorio que se esta recorriendo
 *  dirname: nombre del directorio que se esta recorriendo
 *  fd: file descriptor del archivo .mytar que se esta creando
 *      instructions: Estructura que contiene la informacion de las opciones de
 *          mytar.
 */
void traverseDir(DIR *dir, char *dirname, int fd, mytar_instructions inst) ;


/* createMyTar
 * --------------
 * Se utiliza para crear el archivo .mytar. Funciona procesando cada
 * archivo recibido en la linea de comandos , utilizando sus atributos
 * para almacenar metada, al igual que su contenido.
 *
 *
 *  files: Archivos a procesar
 *  n_files: Numero de archivos a procesar
 *  instructions: Estructura que contiene la informacion de las opciones de
 *          mytar.
 */
int createMyTar(int n_files, char**files, mytar_instructions inst);

#endif


```