UNIVERSIDAD DE MÁLAGA | uma.es

Dpto. de Lenguajes y Ciencias de la Computación

Programación de Sistemas y Concurrencia

Control Bloque 1 -Temas 1 y 2,
Curso 2019 -2020

APELLIDOS_____NOMBRE_____

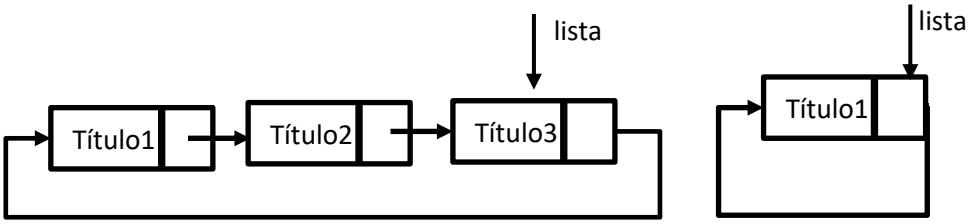
DNI _____ ORDENADOR _____GRUPO /TITULACIÓN_____

Parte 1 (obligatoria para aprobar el control)

El objetivo del ejercicio es simular la gestión en memoria de una lista de reproducción de canciones. Para ello, las canciones se almacenarán en una cola implementada como una lista enlazada circular, tal y como aparece en el dibujo de abajo. La lista de reproducción es un puntero a una estructura definida como sigue:

```
typedef struct Nodo *ListaRep;
struct Nodo{
        char cancion[30];
        ListaRep sig;
};
```

Observa que, en el dibujo, el puntero externo a la estructura apunta al último nodo de la cola, de manera que es posible acceder fácilmente tanto al último (listaRep) y como al primer nodo de la lista (listaRep->sig). En esta implementación una lista vacía es un puntero a NULL, y en una lista con un único nodo, su campo siguiente apunta a sí mismo.



Utilizando la definición de tipos anterior, en la primera parte hay que implementar las siguientes funciones del módulo ListaReproduccion.h en el fichero ListaReproduccion.c

```
/* Crea una lista de reproducción vacía */
void crear(ListaRep *lista);

/* Libera la memoria asociada a la lista de reproducción */
void borrar(ListaRep *lista);
```

```
/* Inserta una nueva canción al final de la lista de reproducción */
void insertar(ListaRep *lista, char * cancion);

/* Muestra por pantalla la lista de canciones almacenada */
void mostrar(ListaRep lista);
```

Parte 2: permite llegar al notable. Sólo se evalúa si se ha aprobado

```
/* Elimina de la lista la canción cuyo título coincide con el parámetro de la función. Devuelve 0 si la canción no está en la lista, y 1 en otro caso */
int eliminar(ListaRep *lista, char * cancion);

/* Devuelve en canción el título de la primera canción que está en la lista de reproducción, y mueve el puntero externo de la lista a la siguiente canción. Devuelve 0 si la lista está vacía, y 1, en otro caso */
int siguiente(ListaRep *lista, char * cancion);
```

Parte 3: permite llegar al sobresaliente - matricula. Sólo se evalúa si se ha llegado al notable

```
/* Guarda en el fichero de texto nombreF la lista de canciones. Cada línea del fichero contiene el título de una canción y aparecerán en el orden de reproducción */
void guardarListaCanciones(ListaRep lista, char *nombreF);

/* Lee del fichero de binario nombreF la lista de canciones y la almacena en la lista de reproducción lista. */
void cargarListaCanciones(ListaRep *lista, char *nombreF);
```

La función **cargarListaCanciones** lee del fichero **binario** de entrada la lista de canciones y las inserta en la lista de reproducción lista. El fichero **binario** tiene como formato una secuencia de pares del tipo <tam tíituto><titulo canción>, donde <tam título> es un entero (int) con la longitud del string <titulo canción> que viene a continuación. Hay que tener en cuenta que la longitud del string es el número de caracteres que preceden al carácter 0 (‘\0’) en la cadena, y que el carácter ‘\0’ está al final de cada uno de los títulos. Por ejemplo, si <5><Te vi> fuera uno de los pares almacenados en el fichero, para leer el string “Te vi” habría que leer 6 caracteres, los 5 caracteres de la cadena, más

uno adicional para el carácter 0. Se asume que los strings del fichero pueden almacenarse en un array de 30 caracteres.

INSTRUCCIONES: descarga el fichero ListaCanciones.bin, ListaReproduccion.h y Driver.c del campus virtual. En Eclipse, crea un nuevo proyecto de C, y copia los ficheros al proyecto. El fichero ListaCanciones.bin es un fichero binario con una lista de canciones con el formato descrito arriba. Realiza la entrega de cada una de las partes en la tarea correspondiente.

Anexo. Los prototipos de las funciones para manipular strings (están en <string.h>) son:

char* strcpy(char *s1, char *s2): Copia los caracteres de la cadena s2 (hasta el carácter '\0', incluido) en la cadena s1. El valor devuelto es la cadena s1.

int strcmp(char *s1, char *s2): Devuelve 0 si las dos cadenas son iguales, <0 si s1 es menor que s2, y >0 si s1 es mayor que s2.

size_t strlen(const char *s): Calcula el número de caracteres de la cadena apuntada por s. *Esta función no cuenta el carácter '\0' que finaliza la cadena*.

Los prototipos de las funciones de lectura y escritura en ficheros de la biblioteca <stdio.h> son los siguientes (se dan por conocidos los prototipos de las funciones de <stdlib.h> que necesites, como free o malloc):

FILE *fopen(const char *path, const char *mode): Abre el fichero especificado en el modo indicado ("rb"/ y "wb" para lectura/escritura binaria y "rt"/"wt" para lectura/escritura de texto). Devuelve un puntero al manejador del fichero en caso de éxito y NULL en caso de error.

int fclose(FILE *fp): Guarda el contenido del buffer y cierra el fichero especificado. Devuelve 0 en caso de éxito y -1 en caso de error.

LECTURA / ESCRITURA BINARIA
unsigned fread(void *ptr, unsigned size, unsigned nmemb, FILE *stream): Lee nmemb elementos de datos, cada uno de tamaño size bytes, desde el fichero stream, y los almacena en la dirección apuntada por ptr. Devuelve el número de elementos leídos.

unsigned fwrite(const void *ptr, unsigned size, unsigned nmemb, FILE *stream): Escribe nmemb elementos de datos, cada uno de tamaño size, al fichero stream, obteniéndolos desde la dirección apuntada por ptr. Devuelve el número de elementos escritos.
LECTURA/ESCRITURA TEXTO
int fscanf(FILE *stream, const char *format, ...): Lee del fichero stream los datos con el formato especificado en el parámetro format, el resto de parámetros son las variables en las que se almacenan los datos leídos en el formato correspondiente. La función devuelve el número de variables que se han leído con éxito.

int fprintf(FILE *stream, const char *format, …): Escribe en el fichero stream los datos con el formato especificado en el parámetro format. El resto de parámetros son las variables en las que se almacenan los datos que hay que escribir. La función devuelve el número de variables que se han escrito con éxito.