ARCHIVOS DIRECTOS / EJERCICIO 6-10

Modifica el programa de la **agenda de contactos** que hicimos en el tema 5. El programa debe disponer ahora de un **menú de opciones** que le permita:

1. ***Añadir nuevos contactos***. *Asígnale un número de orden a cada contacto. El primero será el 1, el segundo el 2, etc. Añade ese número a la estructura de datos del contacto. Luego, graba los contactos en un archivo binario.*
2. ***Listar todos los contactos***. *Hay que recorrer el archivo desde el principio, mostrando en la pantalla toda la información guardada en él.*
3. ***Modificar un contacto***. *El programa nos pedirá un nombre y buscará los contactos que coincidan con ese nombre. Si encuentra alguno, pedirá al usuario que introduzca por teclado los nuevos datos de ese contacto y los grabará en el archivo, sustituyendo a los datos que hubiera antes.*
4. ***Borrar un contacto***. *El programa pedirá un nombre y buscará los contactos que coincidan con él, eliminándolos del archivo mediante el procedimiento de marcarlos como borrados (no borrándolos físicamente)*
5. ***Buscar un contacto***. *Existirán dos formas de búsqueda: por nombre (búsqueda secuencial) o por número de contacto (búsqueda directa)*

SOLUCIÓN

```
#include <stdio.h>

// Definición de constantes
#define INSERTAR 1                      // Opciones del menú
#define LISTAR 2
#define MODIFICAR 3
#define BORRAR 4
#define BUSCAR_NOMBRE 5
#define BUSCAR_NUMERO 6
#define SALIR 7

#define ARCHIVO_DATOS "6_10.dat"        // Nombre del archivo de datos

// Definición de tipos de datos
struct s_contacto
{
   char nombre[100];
   char telef[10];
   char direc[100];
   char email[100];
   char borrado;                        // Marca para saber si el registro ha sido borrado
};

// Prototipos de funciones
void borrar_pantalla(void);
void pulsar_tecla(void);
int menu(void);
void insertar_datos(void);
void listar_datos(void);
void borrar_registro(void);
void modificar_registro(void);
void buscar_por_nombre(void);
void buscar_por_numero(void);
void introducir_datos_contacto(struct s_contacto* contacto);


int main(void)
{
   int opc;

   do
   {
      borrar_pantalla();
      opc = menu();
```

```
      switch (opc)
      {
         case INSERTAR: insertar_datos(); break;
         case LISTAR: listar_datos(); break;
         case MODIFICAR: modificar_registro(); break;
         case BORRAR: borrar_registro(); break;
         case BUSCAR_NOMBRE: buscar_por_nombre(); break;
         case BUSCAR_NUMERO: buscar_por_numero(); break;
      }
   }
   while (opc != SALIR);
}


// Borra la pantalla de texto
void borrar_pantalla(void)
{
   int i;
   for (i=0; i<50; i++)
      printf("\n\n");
}


// Muestra el texto "pulse una tecla para continuar" y espera a que el usuario pulse una tecla
void pulsar_tecla(void)
{
   printf("Pulse Intro para continuar...\n");
   getchar();
}


// Muestra el menú y lee la opción del usuario. Devuelve el código de la opción elegida.
int menu(void)
{
   int opc;
   char aux[50];

   // Muestra el menú
   printf("+----------------------------------+\n");
   printf("|   AGENDA DE CONTACTOS - MENÚ     |\n");
   printf("+----------------------------------+\n");
   printf("|      1. Insertar contacto        |\n");
   printf("|      2. Listar agenda            |\n");
   printf("|      3. Modificar contacto       |\n");
   printf("|      4. Borrar contacto          |\n");
   printf("|      5. Buscar por nombre        |\n");
   printf("|      6. Buscar por número        |\n");
   printf("|      7. Salir                    |\n");
   printf("+----------------------------------+\n");

   // Leer y devolver la selección del usuario
   do
   {
      printf("       Elija una opción (1-7): ");
      gets(aux);
      opc = atoi(aux);
   }
   while ((opc < 1) && (opc > 7));

   return opc;
}
```

```
// Añade un contacto al archivo de datos
void insertar_datos(void)
{
   FILE *f;
   struct s_contacto nuevo;

   // Introducir por teclado los datos del nuevo contacto
   borrar_pantalla();
   printf("INSERTAR NUEVO CONTACTO\n");
   printf("Introduzca los datos del contacto.\n");
   introducir_datos_contacto(&nuevo);

   // Abrir el archivo para añadir datos en él (modo secuencial, para agregar al final)
   f = fopen(ARCHIVO_DATOS, "ab");
   fseek(f, 0L, SEEK_END);
   if (f == NULL)
   {
      printf("Error al abrir el archivo de datos!");
      pulsar_tecla();
   }
   else
   {
      // Escribe el registro en el archivo y lo cierra
      fwrite(&nuevo, sizeof(struct s_contacto), 1, f);
      fclose(f);
      printf("Contacto insertado con éxito\n");
      pulsar_tecla();
   }
}


// Muestra por la pantalla una lista con todos los datos del archivo de datos
void listar_datos(void)
{
   FILE *f;
   struct s_contacto contacto;
   int n;

   // Abre el archivo en modo de lectura secuencial
   f = fopen(ARCHIVO_DATOS, "rb");
   if (f == NULL)
   {
      printf("Error al abrir el archivo de datos");
      pulsar_tecla();
   }
   else
   {
       // Escribe en la pantalla la cabecera de la lista
       borrar_pantalla();
       printf("NOMBRE               DIRECCIÓN            TELÉFONO    E-MAIL\n");
       printf("--------------------------------------------------------------------------------\n");
       while (!feof(f))
       {
           // Lee un registro del archivo
           n = fread(&contacto, sizeof(struct s_contacto), 1, f);
           // Muestra por la pantalla el contenido del registro (si no está marcado como borrado)
           if ((n>0) && (contacto.borrado == 'N'))
               printf("%-20s %-20s %-11s %-15s\n", contacto.nombre, contacto.direc, contacto.telef, contacto.email);
       } // while
       fclose(f);
       pulsar_tecla();
   } // else
}
```

```
// Borra un registro del archivo de datos. En realidad, el registro no se borra, sino
// que se le cambia el valor del campo "borrado"
void borrar_registro(void)
{
   FILE *f;
   struct s_contacto contacto;
   int encontrado, num_reg, n;
   char nombre[100];

   // Leer por teclado el nombre que se quiere borrar
   borrar_pantalla();
   printf("BORRAR REGISTRO\n");
   printf("Introduzca el nombre del contacto que desea borrar: ");
   gets(nombre);

   // Abrimos el archivo en modo directo
   f = fopen(ARCHIVO_DATOS, "r+b");
   if (f == NULL)
   {
      printf("Error al abrir el archivo de datos\n");
      pulsar_tecla();
   }
   else
   {
      // Recorre el archivo de forma secuencial buscando el registro que se quiere borrar
      encontrado = 0;
      num_reg = 0;
      while (!feof(f))
      {
         n = fread(&contacto, sizeof(struct s_contacto), 1, f);
         if (n>0) num_reg++;                              // Contador del nº de registros leidos
         if ((n>0) && (strcmp(contacto.nombre, nombre) == 0))   // Hemos encontado el registro
         {
            // Hemos encontrado el registro que hay que borrar.
            encontrado++;
            // Lo marcamos como borrado
            contacto.borrado = 'S';
            // Colocamos el cursor justo al principio del registro actual
            fseek(f, (num_reg - 1) * sizeof(struct s_contacto), SEEK_SET);
            // Escribimos el registro (ya marcado como borrado) en el archivo
            fwrite(&contacto, sizeof(struct s_contacto), 1, f);
            // Volvemos a dejar el cursor al final del registro
            fseek(f, num_reg * sizeof(struct s_contacto), SEEK_SET);
         }
      }
      fclose(f);
      printf("Se econtraron y borraron %i registros\n", encontrado);
      pulsar_tecla();
   } //else
}


// Modifica un registro del archivo de datos
void modificar_registro(void)
{
   FILE *f;
   struct s_contacto contacto;
   int encontrado, num_reg, n;
   char nombre[100];

   // Leer por teclado el nombre que se quiere borrar
   borrar_pantalla();
   printf("MODIFICAR REGISTRO\n");
   printf("Introduzca el nombre del contacto que desea modificar: ");
   gets(nombre);
```

```
   // Abrimos el archivo en modo directo
   f = fopen(ARCHIVO_DATOS, "r+b");
   if (f == NULL)
   {
      printf("Error al abrir el archivo de datos\n");
      pulsar_tecla();
   }
   else
   {
      // Recorre el archivo de forma secuencial buscando el registro que se quiere modificar
      encontrado = 0;
      num_reg = 0;
      while (!feof(f))
      {
         n = fread(&contacto, sizeof(struct s_contacto), 1, f);
         if (n>0) num_reg++;                                  // Contador del nº de registros leidos
         if ((n>0) && (strcmp(contacto.nombre, nombre) == 0))   // Hemos encontado el registro
         {
            // Hemos encontrado el registro que hay que modificar.
            encontrado++;
            // Le pedimos al usuario que vuelva a introducir los datos por teclado
            printf("Registro encontrado. Introduzca los nuevos datos:\n");
            introducir_datos_contacto(&contacto);
            // Colocamos el cursor justo al principio del registro actual
            fseek(f, (num_reg - 1) * sizeof(struct s_contacto), SEEK_SET);
            // Escribimos la estructura (que ya contiene los datos nuevos) en el archivo
            fwrite(&contacto, sizeof(struct s_contacto), 1, f);
            // Volvemos a dejar el cursor al final del registro
            fseek(f, num_reg * sizeof(struct s_contacto), SEEK_SET);
         }
      }
      fclose(f);
      printf("Se modificaron %i registros\n", encontrado);
      pulsar_tecla();
   } //else
}


// Busca un registro por nombre (búsqueda secuencial)
void buscar_por_nombre(void)
{
   char nombre[100];
   struct s_contacto contacto;
   FILE *f;
   int encontrado, n;

   // Lee el nombre buscado por teclado
   printf("BÚSQUEDA POR NOMBRE\n");
   printf("Introduzca el nombre que quiere buscar: ");
   gets(nombre);

   // Abre el archivo para lectura secuencial
   f = fopen(ARCHIVO_DATOS, "rb");
   if (f == NULL)
   {
      printf("Error al abrir el archivo\n");
      pulsar_tecla();
   }
   else
   {
       encontrado = 0;
       while (!feof(f))
       {
          n = fread(&contacto, sizeof(struct s_contacto), 1, f);
          if ((n>0) && (strcmp(contacto.nombre, nombre) == 0) && (contacto.borrado == 'N'))
          {  // Hemos encontrado el registro
```

```
            // Mostrar datos del registro por la pantalla
            encontrado = 1;
            printf("REGISTRO ENCONTRADO:\n");
            printf("Nombre: %s\n", contacto.nombre);
            printf("Dirección: %s\n", contacto.direc);
            printf("Teléfono: %s\n", contacto.telef);
            printf("e-mail: %s\n", contacto.email);
            pulsar_tecla();
          }
       }
       fclose(f);
       if (encontrado == 0)
       {
         printf("Registro no encontrado.\n");
         pulsar_tecla();
       }
    }
}


// Busca un registro por número (búsqueda directa)
void buscar_por_numero(void)
{
   int numero;
   char aux[50];
   struct s_contacto contacto;
   FILE *f;
   int encontrado, n;

   // Lee el nombre buscado por teclado
   printf("BÚSQUEDA POR NÚMERO\n");
   printf("Introduzca el número del registro que quiere buscar: ");
   gets(aux);
   numero = atoi(aux);

   // Abre el archivo para acceso directo
   f = fopen(ARCHIVO_DATOS, "r+b");
   if (f == NULL)
   {
      printf("Error al abrir el archivo\n");
      pulsar_tecla();
   }
   else
   {
       encontrado = 0;
       // Coloca el cursor en la posición del registro solicitado
       fseek(f, (numero - 1) * sizeof(struct s_contacto), SEEK_SET);
       // Lee el registro situado en esa posición
       n = fread(&contacto, sizeof(struct s_contacto), 1, f);
       // Comprueba que se haya leído algo
       if ((n>0) && (contacto.borrado == 'N'))   // Hemos encontrado el registro
       {
             // Mostrar datos del registro por la pantalla
             encontrado = 1;
             printf("REGISTRO ENCONTRADO:\n");
             printf("Nombre: %s\n", contacto.nombre);
             printf("Dirección: %s\n", contacto.direc);
             printf("Teléfono: %s\n", contacto.telef);
             printf("e-mail: %s\n", contacto.email);
             pulsar_tecla();
       }
       fclose(f);
       if (encontrado == 0)
       {
         printf("Registro no encontrado.\n");
         pulsar_tecla();
```

```
       }
    }
}


// Lee por teclado los datos de un contacto y los devuelve en el parámetro pasado por variable
void introducir_datos_contacto(struct s_contacto* contacto)
{
   printf("  Nombre: ");
   gets(contacto->nombre);
   printf("  Dirección: ");
   gets(contacto->direc);
   printf("  Teléfono: ");
   gets(contacto->telef);
   printf("  E-mail: ");
   gets(contacto->email);
   // Ponemos la marca de "no borrado"
   contacto->borrado = 'N';
}
```