	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorio de computación Salas A y B

Profesor: Dr. Ismael Everardo Bárcenas Patiño

Asignatura: Estructuras de datos y algoritmos I

Grupo: 3

No. de práctica: 5

Integrantes:
10 Gurrión Aquino Carlos Ángel
14 León Ruiz Eduardo
16 Macías Niño Carmen Violeta
17 Marroquín García Ricardo
21 Montaño Torres Rodolfo Santiago

No. de equipo de cómputo empleado:

No. de brigada: 7

Semestre: 2022-1

Fecha de entrega: 5 de octubre de 2021

Observaciones:

Calificación:

Rúbrica de evaluación:

100	El programa cumple con todos los requerimientos
80-99	El programa cumple con la mayoría de los requerimientos
60-79	El programa cumple con algunos de los requerimientos
50-59	La lógica del programa es correcta, pero no corre o se cuelga

Práctica 5 | EDL: Pila y cola

Problema a resolver:

I. Escribir un manejador de base de datos (*playlist*) en lenguaje C con las siguientes características.

- La base de datos está implementada en una pila o en una cola (usando memoria dinámica).
- El manejador tiene las funciones: agregar, borrar y mostrar la base de datos.
- La base de datos contiene los siguientes registros: nombre de la canción, autor, género, valoración.

Código solución:

Enlace al archivo C guardado en nuestro repositorio de GitHub:
https://github.com/Rodolfo-Santiago/EDA1-2022-1-Practicas/blob/main/codes/eda1_p5_code_versionfinal1.c

Enlace al código en un compilador en línea:
https://onlinegdb.com/ysPLGThoe

```
#include <stdio.h>
#include <stdlib.h>


struct Canciones
{
	char nombre[100];
	char autor[50];
	char genero[50];
	float valoracion;
};

struct Nodo
{
	struct Canciones cancion;
	struct Nodo* siguiente;
};


struct Nodo* cabecera;


void descripcion_programa( );
void menu_acciones( );
void insertar_cancion( );
void insertar_datos_cancion(struct Nodo*);
void borrar_cancion( );
void imprimir_base_datos( );
void limpiar_bufer( );



int main( )
{
	int opcion;

	cabecera = NULL;

	descripcion_programa( );

	do
	{
		menu_acciones( );

		if (scanf("%d", &opcion) != 1)
		{
			printf("\n\tError: El dato que ingreso es invalido.\n");
			return 0;
		}

		switch (opcion)
		{
```

```
                        case 1:
                                insertar_cancion( );
                                break;

                        case 2:
                                borrar_cancion( );
                                break;

                        case 3:
                                imprimir_base_datos( );
                                break;

                        case 4:
                                while(cabecera != NULL)
                                {
                                        borrar_cancion( );
                                }
                                printf("\n\nBase de datos descartada y programa finalizado con exito.\n");
                                return 0;

                        default:
                                printf("\n\tOpcion invalida. Digite una de las cuatro opciones determinadas.\n\n");
                }
        } while (1);

        return 0;
}



void descripcion_programa( )
{
        printf("BASE DE DATOS DE CANCIONES\n");
        printf("\nEn esta base de datos el usuario puede almacenar los datos de cualquier ");
        printf("cantidad de canciones. La base de datos esta vacia al comenzar el programa.");
        printf(" El usuario debe elegir que accion realizar dentro del siguiente menu.");
}



void menu_acciones( )
{
        printf("\n\nMenu de acciones:\n\t1. Registrar una nueva cancion en la base de datos");
        printf("\n\t2. Borrar la ultima cancion ingresada en la base de datos\n\t");
        printf("3. Imprimir todas las canciones almacenadas en la base de datos\n\t");
        printf("4. Salir de la base de datos\n\n\tElija la accion que desea ejecutar: ");
}



void insertar_cancion( )
{
        struct Nodo* temp;
        temp = (struct Nodo*) calloc(1, sizeof(struct Nodo));

        insertar_datos_cancion(temp);

        (*temp).siguiente = cabecera;

        cabecera = temp;
}
```

```
void insertar_datos_cancion(struct Nodo* temp)
{
	printf("\n\nREGISTRO DE UNA NUEVA CANCION\n");

	limpiar_bufer( );
	printf("\n\tIntroduzca el nombre de la cancion: ");
	fgets((*temp).cancion.nombre, 100, stdin);

	printf("\tIntroduzca el autor: ");
	fgets((*temp).cancion.autor, 50, stdin);

	printf("\tIntroduzca el genero: ");
	fgets((*temp).cancion.genero, 50, stdin);

	printf("\tIntroduzca la calificacion (0.0-10.0): ");
	if (scanf("%f", &(*temp).cancion.valoracion) != 1)
	{
		printf("\nError: El dato que ingreso es invalido.\n");
		exit(-1);
	}

	while ((*temp).cancion.valoracion < 0.0 || (*temp).cancion.valoracion > 10.0)
	{
		printf("\tCalificacion inválida. Ingrese otra calificacion: ");
		scanf("%f", &(*temp).cancion.valoracion);
	}
}



void borrar_cancion( )
{
	if (cabecera == NULL)
	{
		printf("\n\tError: La base de datos se encuentra vacia. No hay canciones que borrar.\n");
		return;
	}

	struct Nodo* temp;
	temp = cabecera;

	cabecera = (*cabecera).siguiente;

	free(temp);
}



void imprimir_base_datos( )
{
	struct Nodo* auxiliar;

	if (cabecera == NULL)
	{
		printf("\n\tError: La base de datos se encuentra vacia. No hay canciones que mostrar.\n");
		return;
	}

	printf("\n\nIMPRESION DE LAS CANCIONES EN LA BASE DE DATOS\n");
	printf("\nLa lista de todas las canciones es:");

	auxiliar = cabecera;
```

```
	while (auxiliar != NULL)
	{
		printf("\n\tNombre: %s", (*auxiliar).cancion.nombre);
		printf("\tAutor: %s", (*auxiliar).cancion.autor);
		printf("\tGenero: %s", (*auxiliar).cancion.genero);
		printf("\tValoracion: %.1f\n", (*auxiliar).cancion.valoracion);

		auxiliar = (*auxiliar).siguiente;
	}
}



void limpiar_bufer( )
{
	while (fgetc(stdin) != '\n') ;
}
```