En C, las cadenas son arreglos de caracteres terminados en '\0'. Para leer texto de forma segura, se recomienda usar fgets() en lugar de scanf() y eliminar el salto de línea con strcspn(). Estructura Estudiante (basada en el documento):

Trabajaremos con estudiantes.txt. Cada línea se leerá con fgets() y se convertirá a un Estudiante con sscanf(), usando el mismo patrón del documento base.}

La operación Crear solicita datos y los guarda en modo "a" (añadir) usando fprintf(). Antes de agregar, es buena práctica validar que el ID no exista.

Leer consiste en abrir el archivo en modo "r" y mostrar los registros. Se lee línea por línea con fgets().

Buscar es una consulta específica: recorrer el archivo y comparar IDs con strcmp().

En archivos de texto secuenciales, una estrategia práctica para actualizar es: leer el archivo original y escribir uno temporal. Si la línea corresponde al ID buscado, se escribe la versión actualizada; de lo contrario, se copia la línea tal cual. Al final, se reemplaza el archivo original por el temporal. Este patrón es análogo al ejemplo del documento base para eliminar: usar tmp.txt, luego remove() y rename().

Eliminar (baja física) se implementa igual que en el ejemplo del documento base: se lee el archivo original y se reescribe un temporal omitiendo el registro a eliminar. Luego se reemplaza el original.

Integra las funciones en un ciclo do-while. Para primer nivel, un menú por consola es suficiente.

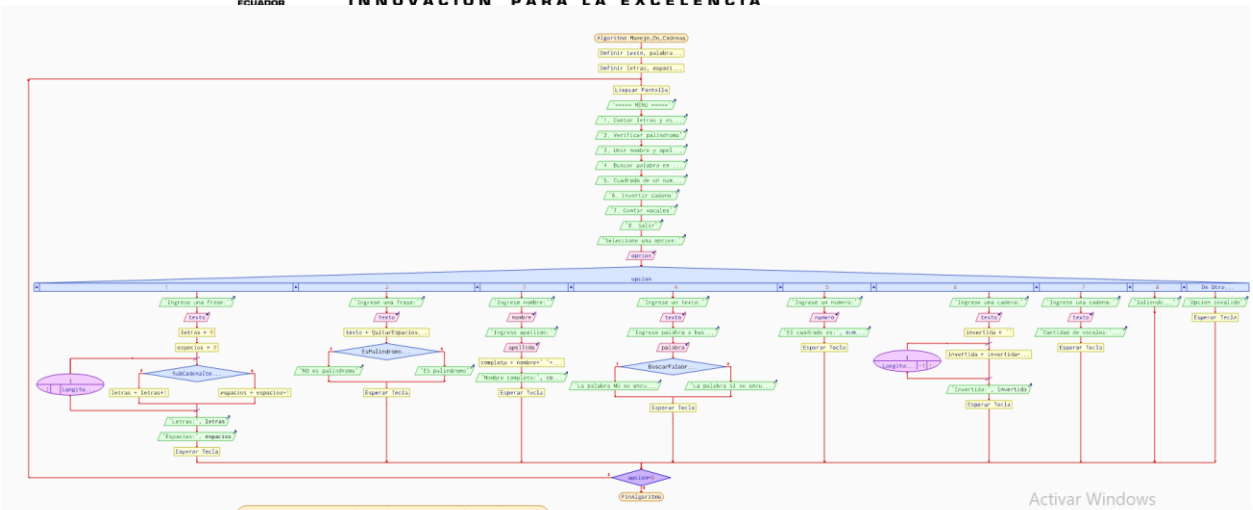
```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

#define ARCHIVO_EST "estudiantes.txt"

/* ===== ESTRUCTURA ===== */
Typedef struct {
    Char ID[20];
    Char apellidos[50];
    nombres de caracteres[50];
    int edad;
} Estudiante;

/* ===== UTILIDADES ===== */
void limpiarNuevaLinea(char *s) {
    s[strcspn(s, "\n")] = '\0';
}

void leerCadena(const char *msg, char *dest, int tam) {
    printf("%s", MSG);
    fgets(dest, tam, stdin));
    limpiarNuevaLinea(dest);
}

int leerEntero(const char *msg) {
    Char Buf[50];
    char *endptr;
    val largo ;

    mientras (1) {
        printf("%s", MSG);
        fgets(buf, sizeof(buf), stdin);
        val = strtol(buf, & endptr, 10);

        si (endptr != buf && *endptr == '\n') {
            return (int)val;
        }
        printf("Entrada inválida. Intente nuevamente.\n");
    }
}

int parsearEstudiante(const char *linea, Estudiante *e) {
    return sscanf(linea, "%19[^;];%49[^;];%49[^;];%d",
                  e->id, e->apellidos, e->nombres, &e->edad) == 4;
}

/* ===== CRUD ===== */
int existeId(const char *idBuscado) {
    ARCHIVO *f = fopen(ARCHIVO_EST, "r");
    si (!f) retorno 0;

    Char Linea[200];
    Estudiante e;

    mientras que (fgets(linea, sizeof(linea), f)) {
        si (parsearEstudiante(linea, &e) && strcmp(e.id, idBuscado) == 0) {
            fclose(f);
            Retorno 1;
        }
    }
    fclose(f);
    retorno 0;
}
```

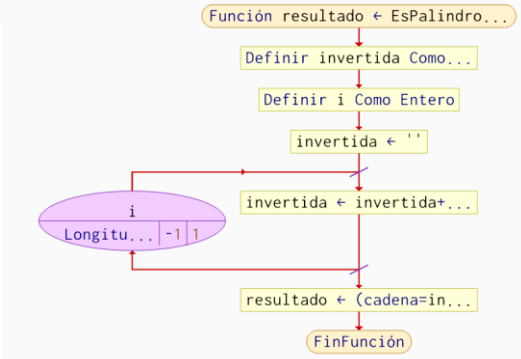
```c
}

void agregarEstudiante() {
    Estudiante e;

    leerCadena("ID: ", e.id, tamaño(e.id));
    si (existeId(e.id)) {
        Printf("El ID ya existe.\n");
        Regreso;
    }

    leerCadena ("Apellidos: ", e.apellidos, tamaño de(e.apellidos));
    leerCadena("Nombres: ", e.nombres, tamaño(e.nombres));
    e.edad  = leerEntero("Edad: ");

    ARCHIVO *f = fopen(ARCHIVO_EST, "a");
    si (!f) {
        printf("Error al abrir archivo.\n");
        Regreso;
    }

    fprintf(f, "%s;%s;%s;%d\n", e.id, e.Apellidos, E.nombres, por ejemplo.edad);
    fclose(f);
    printf("Estudiante agregado correctamente.\n");
}

void listarEstudiantes() {
    ARCHIVO *f = fopen(ARCHIVO_EST, "r");
    si (!f) {
        printf("No existen registros.\n");
        Regreso;
    }

    Char Linea[200];
    Estudiante e;

     printf("\n%-10s %-20s %-20s %-5s\n", "ID", "APELLIDOS", "NOMBRES", "EDAD");
    printf("----------------------------------------------------------------\n");

    mientras que (fgets(linea, sizeof(linea), f)) {
        si (parsearEstudiante(linea, &e)) {
            printf("%-10s %-20s %-20s %-5d\n",
                   e.id, e.Apellidos, E.nombres, por ejemplo.edad);
        }
    }
    fclose(f);
}

void consultarEstudiante() {
    Char ID[20];
    Estudiante e;

    leerCadena("ID a buscar: ", id, sizeof(id));

    ARCHIVO *f = fopen(ARCHIVO_EST, "r");
    si (!f) {
        printf("Archivo no existe.\n");
        Regreso;
    }

    Char Linea[200];
    mientras que (fgets(linea, sizeof(linea), f)) {
        si (parsearEstudiante(linea, &e) && strcmp(e.id, id) == 0) {
            printf("Encontrado: %s %s - Edad %d\n",
                   e.nombres, por ejemplo.Apellidos, E.edad);
            fclose(f);
            Regreso;
```

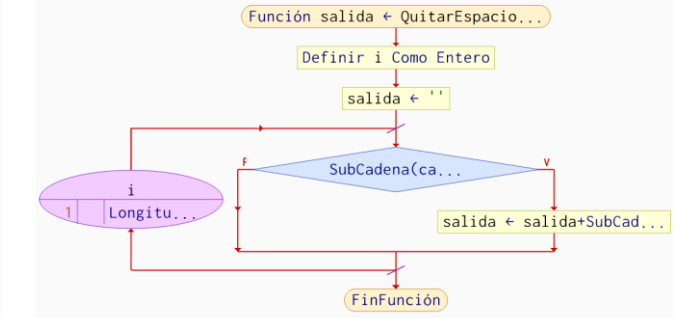
```c
                eliminado = 1;
            }
        }
    }

    fclose(f);
    Fclose (temp);
    eliminar(ARCHIVO_EST);
    renombrar ("tmp.txt", ARCHIVO_EST);

    si (eliminado)
        printf("Registro eliminado.\n");
    si no,
        printf("ID no encontrado.\n");
}

/* ===== MENÚ ===== */
Menú INT Menú() {
    char OP[10];
    printf("\n=== CRUD ESTUDIANTES ===\n");
    printf("1. Agregar\n");
    Printf("2. Listar\n");
    printf("3. Consultor");
    Printf("4. Actualizar\n");
    Printf("5. Eliminar\n");
    printf("0. Salir\n");
    printf("Opción: ");
    fgets(OP, Sizeof(OP), Stdin);
    return atoi(op);
}

Int Main() {
    int op;
    do {
        OP = menú();
        Switch (op) {
            caso 1: agregarEstudiante();  Pausa;
            caso 2: listarEstudiantes();  Pausa;
            caso 3: consultarEstudiante();  Pausa;
            caso 4: actualizarEstudiante();  Pausa;
            caso 5: eliminarEstudiante();  Pausa;
            caso 0: printf("Saliendo...\n");  Pausa;
            default: printf("Opción inválida.\n");
        }
    } mientras (op != 0);

    retorno 0;
}
```

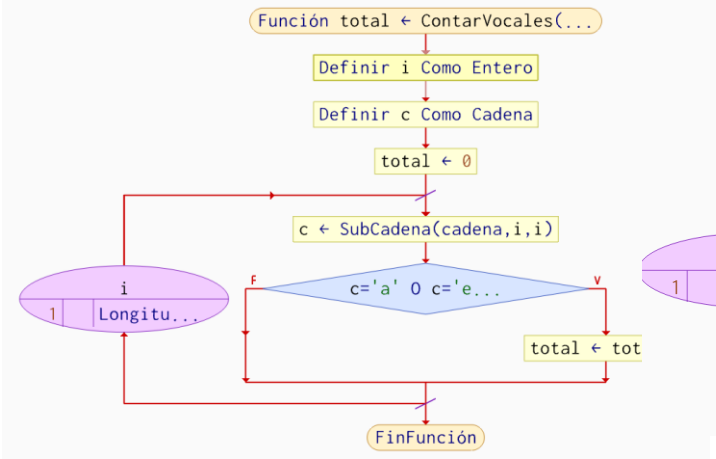
```c
    }
    fclose(f);
    printf("ID no encontrado.\n");
}

void actualizarEstudiante() {
    Char ID[20];
    leerCadena("ID a actualizar: ", id, sizeof(id));

    ARCHIVO *f = fopen(ARCHIVO_EST, "r");
    ARCHIVO *temp = fopen("tmp.txt", "w");
    si (!f ||  ¡Temp) {
        Printf("Error con archivos.\n");
        Regreso;
    }

    Char Linea[200];
    Estudiante e;
    int encontrado = 0;

    mientras que (fgets(linea, sizeof(linea), f)) {
        si (parsearEstudiante(linea, &e) && strcmp(e.id, id) == 0) {
            leerCadena ("Nuevos apellidos: ", e.apellidos, tamaño de(e.apellido
            leerCadena("Nuevos nombres: ", e.nombres, tamaño(e.nombres));
            e.edad  = leerEntero("Nueva edad: ");
            Encontrado = 1;
        }
        fprintf(temp, "%s;%s;%s;%d\n",
                e.id, e.Apellidos, E.nombres, por ejemplo.edad);
    }

    fclose(f);
    Fclose (temp);
    eliminar(ARCHIVO_EST);
    renombrar ("tmp.txt", ARCHIVO_EST);

    si (encontrado)
        printf("Registro actualizado.\n");
    si no,
        printf("ID no encontrado.\n");
}

void eliminarEstudiante() {
    Char ID[20];
    leerCadena("ID a eliminar: ", id, sizeof(id));

    ARCHIVO *f = fopen(ARCHIVO_EST, "r");
    ARCHIVO *temp = fopen("tmp.txt", "w");
    si (!f ||  ¡Temp) {
        printf("Error con archivos.\n");
        Regreso;
    }

    Char Linea[200];
    Estudiante e;
    int eliminado = 0;

    mientras que (fgets(linea, sizeof(linea), f)) {
        si (parsearEstudiante(linea, &e)) {
            si (strcmp(e.id, id) != 0) {
                fprintf(temp, "%s;%s;%s;%d\n",
                        e.id, e.Apellidos, E.nombres, por ejemplo.edad);
            } si no, {
                eliminado = 1;
```

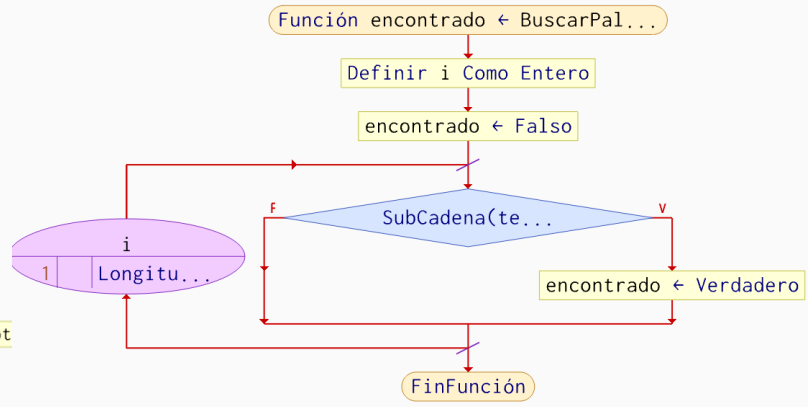
```
Algoritmo Manejo_De_Cadenas

    Definir texto, palabra, nombre, apellido, completo, invertida Como Cadena
    Definir letras, espacios, opcion, numero, i Como Entero

    Repetir
        Limpiar Pantalla
        Escribir "===== MENU ====="
        Escribir "1. Contar letras y espacios"
        Escribir "2. Verificar palindromo"
        Escribir "3. Unir nombre y apellido"
        Escribir "4. Buscar palabra en texto"
        Escribir "5. Cuadrado de un numero"
        Escribir "6. Invertir cadena"
        Escribir "7. Contar vocales"
        Escribir "8. Salir"
        Escribir "Seleccione una opcion:"
        Leer opcion

        Segun opcion Hacer

            1:
                Escribir "Ingrese una frase:"
                Leer texto
                letras ← 0
                espacios ← 0

                Para i ← 1 Hasta Longitud(texto)
                    Si SubCadena(texto,i,i) = " " Entonces
                        espacios ← espacios + 1
                    Sino
                        letras ← letras + 1
                    FinSi
                FinPara

                Escribir "Letras:", letras
                Escribir "Espacios:", espacios
                Esperar Tecla

            2:
                Escribir "Ingrese una frase:"
                Leer texto
                texto ← QuitarEspacios(texto)

                Si EsPalindromo(texto) Entonces
                    Escribir "ES palindromo"
                Sino
                    Escribir "NO es palindromo"
                FinSi
                Esperar Tecla

            3:
                Escribir "Ingrese nombre:"
                Leer nombre
                Escribir "Ingrese apellido:"
                Leer apellido

                completo ← nombre + " " + apellido
                Escribir "Nombre completo:", completo
                Esperar Tecla

            4:
                Escribir "Ingrese un texto:"
                Leer texto
                Escribir "Ingrese palabra a buscar:"
                Leer palabra

                Si BuscarPalabra(texto, palabra) Entonces
                    Escribir "La palabra SI se encuentra"
                Sino
                    Escribir "La palabra NO se encuentra"
                FinSi
                Esperar Tecla

            5:
                Escribir "Ingrese un numero:"
                Leer numero
                Escribir "El cuadrado es:", numero * numero
                Esperar Tecla

            6:
                Escribir "Ingrese una cadena:"
                Leer texto

                invertida ← ""
                Para i ← Longitud(texto) Hasta 1 Con Paso -1
                    invertida ← invertida + SubCadena(texto,i,i)
                FinPara

                Escribir "Invertida:", invertida
                Esperar Tecla

            7:
                Escribir "Ingrese una cadena:"
                Leer texto
                Escribir "Cantidad de vocales:", ContarVocales(texto)
                Esperar Tecla

            8:
                Escribir "Saliendo..."

            De Otro Modo:
                Escribir "Opcion invalida"
                Esperar Tecla

        FinSegun

    Hasta Que opcion = 8

FinAlgoritmo


Funcion resultado ← EsPalindromo(cadena)
    Definir invertida Como Cadena
    Definir i Como Entero

    invertida ← ""
    Para i ← Longitud(cadena) Hasta 1 Con Paso -1
        invertida ← invertida + SubCadena(cadena,i,i)
    FinPara
    resultado ← (cadena = invertida)
FinFuncion


Funcion salida ← QuitarEspacios(cadena)
    Definir i Como Entero
    salida ← ""

    Para i ← 1 Hasta Longitud(cadena)
        Si SubCadena(cadena,i,i) ≠ " " Entonces
            salida ← salida + SubCadena(cadena,i,i)
        FinSi
    FinPara
FinFuncion


Funcion total ← ContarVocales(cadena)
    Definir i Como Entero
    Definir c Como Cadena
    total ← 0

    Para i ← 1 Hasta Longitud(cadena)
        c ← SubCadena(cadena,i,i)
        Si c="a" O c="e" O c="i" O c="o" O c="u" Entonces
            total ← total + 1
        FinSi
    FinPara
FinFuncion


Funcion encontrado ← BuscarPalabra(texto, palabra)
    Definir i Como Entero
    encontrado ← Falso

    Para i ← 1 Hasta Longitud(texto) - Longitud(palabra) + 1
        Si SubCadena(texto,i,i+Longitud(palabra)-1) = palabra Entonces
            encontrado ← Verdadero
        FinSi
    FinPara
FinFuncion
```











https://onlinegdb.com/8X6s1a_LN-

}