

Sistema de Gestión de Estudiante

Nombre Completo del Estudiante: Kehit Nathaniel Fernandez Mori

Nombre de la Institución: Tecnica N°2

Curso: Laboratorio de Programación 5° 2° B

Profesor: Mansilla Muñoz York E.

Fecha: 20/09/2025

Menu Principal:

```
File Edit Selection View Go Run ... ☰ Sobre C:\Users\Estudiante>Documentos>proyectos>Sistema_Gestion_Estudiantes_Maven>src>C.java

1 // Main principal
2
3 int main() {
4     asegurarArchivos();
5
6     int opcion;
7     do {
8         printf("\n--- SISTEMA DE GESTIÓN DE ESTUDIANTES ---\n");
9         printf("1) Registrar estudiante\n");
10        printf("2) Buscar por legajo\n");
11        printf("3) Listar estudiantes\n");
12        printf("4) Modificar estudiante (opcional)\n");
13        printf("5) Baja lógica (recomendada)\n");
14        printf("6) Salir\n");
15        printf("Selección una opción: ");
16        if (scanf("%d", &opcion) != 1) {
17            limpiarBuffer();
18            opcion = -1;
19        }
20        limpiarBuffer();
21
22        switch (opcion) {
23            case 1: registrarEstudiante(); break;
24            case 2: buscarPorLegajo(); break;
25            case 3: listarEstudiantes(); break;
26            case 4: modificarEstudiante(); break;
27            case 5: bajaLogica(); break;
28            case 6: printf("Fin del programa.\n"); break;
29            default: printf("Opción invalida.\n"); break;
30        }
31    } while (opcion != 6);
32 }
```

```
File Edit Selection View Go Run ... D -> C:\Users\Estudiante\Documents\proyectos>Sistemas_Gestion_Estudiantes\main < C:\main

1 #include <stdio.h>
2
3 int main() {
4     do {
5         printf("1) Nuevo estudiante\n");
6         printf("2) Borrar un estudiante\n");
7         printf("3) Listar estudiantes\n");
8         printf("4) Modificar estudiante (selección)\n");
9         printf("5) Baja lógica (selección)\n");
10        printf("6) Salir\n");
11
12        printf("Seleccione una opción: ");
13        if (scanf("%d", &opcion) != 1)
14            llenaPuffer();
15        opcion = -1;
16
17        llenaPuffer();
18
19        switch (opcion) {
20            case 1: registrarEstudiante(); break;
21            case 2: borrarPorId(); break;
22            case 3: listarEstudiantes(); break;
23            case 4: modificarEstudiante(); break;
24            case 5: bajaLogica(); break;
25            case 6: printf("\nAdiós programón"); break;
26            default: printf("Opción inválida\n"); break;
27        }
28    } while (opcion != 0);
29
30    return 0;
31
32 }
```

El programa del menu principal funciona de la siguiente manera: primero, se asegura de que entiendas las opciones que tienes disponibles, mostrandote un menu numerado. Luego, espera pacientemente a que ingreses el numero de la opcion que deseas. Para esto, utiliza la funcion scanf es importante que ingreses un numero, ya que el programa verifica si lo que escribiste es valido. Si te equivocas y escribes algo que no es un numero, el programa lo detecta y te dice que la opcion no es valida. Una vez que el programa tiene tu opcion (y se asegura de que sea valida), utiliza una estructura llamada switch para decidir que accion realizar. Cada numero del menu esta asociado a una accion especifica. Por ejemplo, si eliges la opcion 1, el programa ejecutara la funcion registrarEstudiante ().

Operaciones:

```
File Edit Selection View Go Run ... D menu
C:\Users\Estudiante\Documents\3 proyectos> Sistema.Gestion.Estudantes\main> cd ..> C main.c

1 // 
2 // Operaciones
3 // 
4 void registrarEstudiante() {
5     FILE *f = fopen(ARCHIVO_DATOS, "ab");
6     if (!f) { printf("Error abriendo archivo.\n"); return; }
7 
8     Estudiante e;
9     printf("Nombre: ");
10    scanf("%s", &e.nombre);
11   	fflushBuffer();
12 
13    insertarLinea("Nombre y apellido: ", e.nombre, sizeof(e.nombre));
14 
15    printf("Promedio (0.00 - 10.00): ");
16    scanf("%f", &e.promedio);
17    fflushBuffer();
18 
19    e.activo = 1;
20 
21    size_t escritos = fwrite(&e, sizeof(Estudiante), 1, f);
22    if (escritos == 1) {
23        printf("Estudiante registrado con exito.\n");
24    } else {
25        printf("Error al escribir en el archivo.\n");
26    }
27    fclose(f);
28 }

// Desarrollado por: J.
INCLUDE DEBUG CONSOLE TERMINAL PORT
Legado: 3
Nombre y apellido: asdasd asd asd
++ Bash
[2] gedit ed...
```

```
File Edit Selection View Go Run ... D menu
C:\Users\Estudiante\Documents\3 proyectos> Sistema.Gestion.Estudantes\main> cd ..> C main.c

1 void listarEstudiantes() {
2     FILE *f = fopen(ARCHIVO_DATOS, "rb");
3     if (!f) { printf("Error abriendo archivo.\n"); return; }
4 
5     Estudiante e;
6     int contador = 0;
7     printf("\n--- LISTADO DE ESTUDIANTES ACTIVOS ---\n");
8     while (fread(&e, sizeof(Estudiante), 1, f) == 1) {
9         if (e.activo == 1) {
10            printf("Ingenieria: %s | Nombre: %s | Promedio: %.2f\n", &e.legajo, &e.nombre, &e.promedio);
11            contador++;
12        }
13    }
14    if (contador == 0) printf("(sin estudiantes activos)\n");
15    fclose(f);
16 }

17 void buscarPorLegajo() {
18     FILE *f = fopen(ARCHIVO_DATOS, "rb");
19     if (!f) { printf("Error abriendo archivo.\n"); return; }
20 
21     int legajo;
22     printf("Ingrese legajo a buscar: ");
23     scanf("%d", &legajo);
24     fflushBuffer();
25 
26     Estudiante e;
27     int encontrado = 0;
28     while (fread(&e, sizeof(Estudiante), 1, f) == 1) {
29         if (e.legajo == legajo && e.activo == 1) {
30             encontrado++;
31         }
32     }
33     if (encontrado == 0) printf("No se encontro el legajo.\n");
34     fclose(f);
35 }

INCLUDE DEBUG CONSOLE TERMINAL PORT
Legado: 3
Nombre y apellido: asdasd asd asd
++ Bash
[2] gedit ed...
```

```
File Edit Selection View Go Run ... Prefs
C:\Users\Estudiante>Documents>proyectos>Sistema_Gestion_Estudiantes>main.c C:\main.c
void modificarEstudiante() {
    char linea[100];
    FILE *fp;
    int pos = ftell(fp) == 0 ? fseek(fp, sizeof(Estudiante), 1, fp) == 1 : 1;
    if (escribir == logico && e.activo == 1) {
        printf("Modificar: dejar vacío para no cambiar)\n");
    }
    else {
        char buffer[40];
        leerCadena("Nuevo nombre y apellido: ", buffer, sizeof(buffer));
        if (strlen(buffer) > 0) strcpy(e.nombre, buffer);
        if (e.nombre[0] != '\0') strcpy(e.apellido, buffer);
        else strcpy(e.apellido, e.nombre);
        printf("Nuevo apellido (%c) para su nombre)\n");
        float p;
        if (scanf("%f", &p) == 1 && p >= 0.0f) {
            e.promedio = p;
        }
        limpiarBuffer();
    }
    // valor al inicio del registro y sobreescritura
    fseek(fp, pos, SEEK_SET);
    size_t escritos = fwrite(fp, sizeof(Estudiante), 1, fp);
    if (escritos == 1) {
        printf("Registro modificado.\n");
        modificada = 1;
    } else {
        printf("Error al sobreescibir registro.\n");
    }
    limpiar();
}
if (!modificada) printf("No se ha hecho ninguna acción con logico %d.\n", logico);
}

Process: OUTPUT DEBUG CONSOLE TERMINAL PORTS
Logado: 3
Nombre y apellido: gomez carlos

```

```
File Edit Selection View Go Run ... D:\main.c
C:\Users\Estudiante\Documents\3 proyectos\3 Estudiantes\binarios\binario> cd ..> C:\main
132 void modificarEstudiante() {
133     if (!feof(f)) printf("No se encontro estudiante activo con legajo %d.\n", legajo);
134     fclose(f);
135 }
136 //OPCIONAL Recomendado: Baja logica
137 void bajaLogica() {
138     FILE *f = fopen(ARCHIVO_DATOS, "rb+");
139     if (!f) { printf("Error al abrir archivo.\n"); return; }

140     int legajo;
141     printf("Ingrese legajo a dar de baja: ");
142     scanf("%d", &legajo);
143     limpiarBuffer();
144
145     Estudiante e;
146     long pos;
147     int lejado = 0;
148
149     while ((pos = ftell(f)) >= 0) fread(&e, sizeof(Estudiante), 1, f) == 1) {
150         if (e.legajo == legajo && e.activo == 1) {
151             e.activo = 0;
152             fseek(f, pos, SEEK_SET);
153             if (fwrite(&e, sizeof(Estudiante), 1, f) == 1) {
154                 printf("Se ha logico realizar.\n");
155                 lejado = 1;
156             } else {
157                 printf("Error al actualizar registro.\n");
158             }
159         }
160     }
161
162     if (!lejado) printf("No se encontro estudiante activo con legajo %d.\n", legajo);
163     fclose(f);
164 }
```

Nombre y apellido: aaaaaaaa

```
File Edit Selection View Go Run ... D:\main.c
C:\Users\Estudiante\Documents\3 proyectos\3 Estudiantes\binarios\binario> cd ..> C:\main
132 void modificarEstudiante() {
133     FILE *f = fopen(ARCHIVO_DATOS, "rb+");
134     if (!f) { printf("Error al abrir archivo.\n"); return; }

135     int legajo;
136     printf("Ingrese legajo a dar de baja: ");
137     scanf("%d", &legajo);
138     limpiarBuffer();
139
140     Estudiante e;
141     long pos;
142     int lejado = 0;
143
144     while ((pos = ftell(f)) >= 0) fread(&e, sizeof(Estudiante), 1, f) == 1) {
145         if (e.legajo == legajo && e.activo == 1) {
146             e.activo = 0;
147             fseek(f, pos, SEEK_SET);
148             if (fwrite(&e, sizeof(Estudiante), 1, f) == 1) {
149                 printf("Se ha logico realizar.\n");
150                 lejado = 1;
151             } else {
152                 printf("Error al actualizar registro.\n");
153             }
154         }
155     }
156
157     if (!lejado) printf("No se encontro estudiante activo con legajo %d.\n", legajo);
158     fclose(f);
159 }
```

Nombre y apellido: aaaaaaaa

El Código implementa un sistema de gestión de estudiantes que utiliza un archivo binario para almacenar la información de cada estudiante. Cada estudiante tiene un legajo (ID), nombre, apellido, promedio y un indicador de si está activo o no. las funciones principales permiten registrar nuevos estudiantes, listar los estudiantes activos, buscar un estudiante por su legajo, modificar la información de un estudiante y realizar una baja logica (marcar un estudiante como inactivo). Ademas emplea funciones clave de manejo de archivo: fopen para abrir el archivo en el modo adecuado, fread para leer registros y cargarlos en memoria fwrite para escribir nuevos registros o modificar existentes, fseek para mover el puntero de lectura/escritura a posiciones específicas dentro del archivo y ftell para obtener la posición actual del puntero. El flujo de operaciones incluye abrir el archivo, leer o modificar registros utilizando fseek y las funciones de lectura/escritura, y finalmente cerrar el archivo con fclose, asegurando una gestión eficiente de la información de estudiantes.

Definiciones y Utilitarios:

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

// Definiciones y utilitarios

typedef struct {
    int legajo;
    char nombre[40];
    float promedio;
    int activo; // Activos: 0, Inactivos: 1
} Estudiante;

#define ARCHIVO_DATOS "data/estudiantes.dat"

// Trim del string final de fgets
void trim_lineas(char *s) {
    size_t n = strlen(s);
    if (n > 0 && s[n-1] == '\n') s[n-1] = '\0';
}

// Leer cadena segura
void leerCadena(char *prompt, char *dest, size_t tam) {
    printf("%s", prompt);
    if (fgets(dest, (int)tam, stdin) != NULL) {
        trim_lineas(dest);
    } else {
        // Si falla, deja string en []
        if (tam > 0) dest[0] = '\0';
    }
}

// Limpia el salto de linea posterior luego de scanf
void limpiaBuffer() {
}
```

```
#include "estudiante.h"

Estudiante estudiante;

#define ARCHIVO_DATOS "data/estudiantes.dat"

// Trim del string final de fgets
void trim_lineas(char *s) {
    size_t n = strlen(s);
    if (n > 0 && s[n-1] == '\n') s[n-1] = '\0';
}

// Leer cadena segura
void leerCadena(char *prompt, char *dest, size_t tam) {
    printf("%s", prompt);
    if (fgets(dest, (int)tam, stdin) != NULL) {
        trim_lineas(dest);
    } else {
        // Si falla, deja string en []
        if (tam > 0) dest[0] = '\0';
    }
}

// Limpia el salto de linea posterior luego de scanf
void limpiaBuffer() {
    int c;
    while ((c = getch()) != '\n' && c != EOF) {}
}

// Crear archivo si no existe (modo binario)
void asegurarArchivo() {
    FILE *f = fopen(ARCHIVO_DATOS, "ab");
    if (!f) fclose(f);
}
```

Defini una estructura llamada Estudiante que es como un formulario con los datos básicos: legajo, nombre, promedio y si está activo. Luego, con funciones Como leerCadena para leer nombres de forma segura y limpiarBuffer para evitar errores al leer del teclado. archivo donde guardarán los datos exista, con la función crearArchivo . En conclusión, es un código bien pensado para sentar las bases de un programa que gestione información de estudiantes de forma organizada y sin errores comunes.

Este es el link de github de kehit fernandez mori:

https://github.com/keh19/Sistema_Gestion_Estudiante.git