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

#define MAX_NOMBRE 60
#define MAX_CALLE  60
#define MAX_CIUDAD 40
#define NUM_CALIF  5

typedef struct {
        char calle[MAX_CALLE];
        int  numero;
        char ciudad[MAX_CIUDAD];
} Direccion;

typedef struct {
        char nombre[MAX_NOMBRE];
        int  edad;
        float calificaciones[NUM_CALIF];
        Direccion direccion;
} Estudiante;

void chomp(char *s) {
        size_t n = strlen(s);
        if (n > 0 && s[n-1] == '\n') s[n-1] = '\0';
}

void leerLinea(const char *prompt, char *dest, size_t tam) {
        printf("%s", prompt);
        if (fgets(dest, (int)tam, stdin) == NULL) {
                dest[0] = '\0';
                return;
        }
        chomp(dest);
}

int leerEntero(const char *prompt, int min, int max) {
        char buffer[128];
        long valor;
        char *endptr;
        while (1) {
                printf("%s", prompt);
                if (!fgets(buffer, sizeof(buffer), stdin)) {
                        clearerr(stdin);
                        continue;
                }
```

```
                valor = strtol(buffer, &endptr, 10);
                if (endptr == buffer || (*endptr && *endptr != '\n')) {
                        printf("  Valor no válido. Intenta de nuevo.\n");
                        continue;
                }
                if (valor < min || valor > max) {
                        printf("  Debe estar entre %d y %d.\n", min, max);
                        continue;
                }
                return (int)valor;
        }
}

float leerFloat(const char *prompt, float min, float max) {
        char buffer[128];
        float valor;
        char c;
        while (1) {
                printf("%s", prompt);
                if (!fgets(buffer, sizeof(buffer), stdin)) {
                        clearerr(stdin);
                        continue;
                }
                if (sscanf(buffer, " %f %c", &valor, &c) != 1) {
                        printf("  Valor no válido. Intenta de nuevo.\n");
                        continue;
                }
                if (valor < min || valor > max) {
                        printf("  Debe estar entre %.1f y %.1f.\n", min, max);
                        continue;
                }
                return valor;
        }
}

void aMinusculas(const char *src, char *dst, size_t tam) {
        size_t i;
        for (i = 0; i + 1 < tam && src[i]; ++i) dst[i] = (char)tolower((unsigned char)src[i]);
        dst[i] = '\0';
}

void ingresarEstudiantes(Estudiante **estudiantes, int *cantidad) {
        int agregar = leerEntero("¿Cuántos estudiantes desea ingresar? ", 1, 1000);
        Estudiante *tmp = realloc(*estudiantes, (size_t)(*cantidad + agregar) * sizeof(Estudiante));
        if (!tmp) {
                printf("Error: memoria insuficiente.\n");
                return;
```

```
        }
        *estudiantes = tmp;

        for (int i = 0; i < agregar; ++i) {
                Estudiante e;
                printf("\n--- Estudiante %d ---\n", *cantidad + 1);

                leerLinea("Nombre: ", e.nombre, sizeof(e.nombre));
                e.edad = leerEntero("Edad: ", 1, 120);

                for (int j = 0; j < NUM_CALIF; ++j) {
                        char prompt[64];
                        snprintf(prompt, sizeof(prompt), "Calificación %d (0-10): ", j + 1);
                        e.calificaciones[j] = leerFloat(prompt, 0.0f, 10.0f);
                }

                leerLinea("Calle: ", e.direccion.calle, sizeof(e.direccion.calle));
                e.direccion.numero = leerEntero("Número: ", 0, 1000000);
                leerLinea("Ciudad: ", e.direccion.ciudad, sizeof(e.direccion.ciudad));

                (*estudiantes)[*cantidad] = e;
                (*cantidad)++;
        }
        printf("\nSe ingresaron %d estudiante(s).\n", agregar);
}

void mostrarEstudiantes(const Estudiante *estudiantes, int cantidad) {
        if (cantidad == 0) {
                printf("\nNo hay estudiantes cargados.\n");
                return;
        }
        printf("\n================= LISTA DE ESTUDIANTES =================\n");
        for (int i = 0; i < cantidad; ++i) {
                printf("\n[%d] %s\n", i + 1, estudiantes[i].nombre);
                printf("    Edad: %d\n", estudiantes[i].edad);
                printf("    Calificaciones: ");
                for (int j = 0; j < NUM_CALIF; ++j) {
                        printf("%.1f%s", estudiantes[i].calificaciones[j], (j + 1 < NUM_CALIF) ? ", " : "\n");
                }
                printf("    Dirección: %s %d, %s\n",
                              estudiantes[i].direccion.calle,
                              estudiantes[i].direccion.numero,
                              estudiantes[i].direccion.ciudad);
        }
        printf("\n========================================================\n");
}
```

```
int buscarEstudiante(const Estudiante *estudiantes, int cantidad, const char *nombre) {
        if (cantidad == 0) return -1;
        char needle[MAX_NOMBRE], tmp[MAX_NOMBRE];
        aMinusculas(nombre, needle, sizeof(needle));

        for (int i = 0; i < cantidad; ++i) {
                aMinusculas(estudiantes[i].nombre, tmp, sizeof(tmp));
                if (strstr(tmp, needle)) {
                        return i;
                }
        }
        return -1;
}

void modificarEstudiante(Estudiante *estudiantes, int cantidad) {
        if (cantidad == 0) {
                printf("\nNo hay estudiantes para modificar.\n");
                return;
        }
        char nombre[MAX_NOMBRE];
        leerLinea("\nNombre del estudiante a modificar: ", nombre, sizeof(nombre));
        int idx = buscarEstudiante(estudiantes, cantidad, nombre);
        if (idx == -1) {
                printf("No se encontró el estudiante.\n");
                return;
        }

        Estudiante *e = &estudiantes[idx];
        printf("\nEditando a: %s\n", e->nombre);
        printf("1) Modificar nombre\n");
        printf("2) Modificar edad\n");
        printf("3) Modificar calificaciones\n");
        printf("4) Modificar dirección\n");
        printf("5) Modificar todo\n");
        printf("0) Cancelar\n");

        int op = leerEntero("Seleccione opción: ", 0, 5);
        if (op == 0) return;

        if (op == 1 || op == 5) {
                leerLinea("Nuevo nombre: ", e->nombre, sizeof(e->nombre));
        }
        if (op == 2 || op == 5) {
                e->edad = leerEntero("Nueva edad: ", 1, 120);
        }
        if (op == 3 || op == 5) {
                for (int j = 0; j < NUM_CALIF; ++j) {
```

```
                        char prompt[64];
                        snprintf(prompt, sizeof(prompt), "Nueva calificación %d (0-10): ", j + 1);
                        e->calificaciones[j] = leerFloat(prompt, 0.0f, 10.0f);
                }
        }
        if (op == 4 || op == 5) {
                leerLinea("Nueva calle: ", e->direccion.calle, sizeof(e->direccion.calle));
                e->direccion.numero = leerEntero("Nuevo número: ", 0, 1000000);
                leerLinea("Nueva ciudad: ", e->direccion.ciudad, sizeof(e->direccion.ciudad));
        }

        printf("\nDatos actualizados correctamente.\n");
}

void mostrarMenu(void) {
        printf("\n======= Gestión de Estudiantes =======\n");
        printf("1. Ingresar estudiantes\n");
        printf("2. Mostrar estudiantes\n");
        printf("3. Buscar estudiante\n");
        printf("4. Modificar estudiante\n");
        printf("5. Salir\n");
}

int main(void) {
        Estudiante *estudiantes = NULL;
        int cantidad = 0;
        int opcion;

        do {
                mostrarMenu();
                opcion = leerEntero("Opción: ", 1, 5);

                switch (opcion) {
                        case 1:
                                ingresarEstudiantes(&estudiantes, &cantidad);
                                break;
                        case 2:
                                mostrarEstudiantes(estudiantes, cantidad);
                                break;
                        case 3: {
                                char nombre[MAX_NOMBRE];
                                leerLinea("\nNombre a buscar: ", nombre, sizeof(nombre));
                                int idx = buscarEstudiante(estudiantes, cantidad, nombre);
                                if (idx == -1) {
                                        printf("No se encontró el estudiante.\n");
                                } else {
                                        printf("\nEstudiante encontrado:\n");
```

```
                                        Estudiante *e = &estudiantes[idx];
                                        printf("Nombre: %s\nEdad: %d\n", e->nombre, e->edad);
                                        printf("Calificaciones: ");
                                        for (int j = 0; j < NUM_CALIF; ++j) {
                                                printf("%.1f%s", e->calificaciones[j], (j + 1 < NUM_CALIF) ? ", " : "\n");
                                        }
                                        printf("Dirección: %s %d, %s\n",
                                                        e->direccion.calle, e->direccion.numero, e->direccion.ciudad);
                                }
                                break;
                        }
                        case 4:
                                modificarEstudiante(estudiantes, cantidad);
                                break;
                        case 5:
                                break;
                        default:
                                printf("Opción no válida.\n");
                }
        } while (opcion != 5);

        free(estudiantes);
        printf("\nPrograma finalizado.\n");
        return 0;
}
```