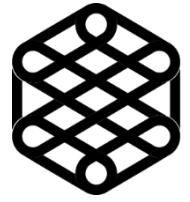




Universidad Autónoma de la Ciudad de México
Nada humano me es ajeno



Licenciatura en Ingeniería de Software
UACM San Lorenzo Tezonco
introducción a la ingeniería en Software

Practica 3 Uso de Arreglos/Buffer, Funciones, apuntadores, memoria dinámica y struc

Ariel Sofia Lopez Amaya
Matricula: 21-003-1193

Introducción

El presente documento tiene como objetivo presentar el desarrollo de un programa en lenguaje **C** que gestiona las **tareas, alumnos y calificaciones**. El sistema permite registrar datos, asignar calificaciones y mostrar resultados de forma estructurada, empleando **estructuras (struct)**, **funciones**, y un **menú principal** para interactuar con el usuario.

Este trabajo busca aplicar los conceptos básicos de **programación modular, uso de arreglos y estructuras**, así como el control de flujo mediante **menús interactivos**. Con ello, se pretende fortalecer la lógica de programación y el diseño ordenado del código fuente.

Desarrollo

Análisis del problema

En muchas situaciones académicas, los docentes y alumnos carecen de un sistema que relacione de manera sencilla las tareas asignadas, los estudiantes y las calificaciones obtenidas.

Tradicionalmente, estos registros se hacen de forma manual o en hojas de cálculo, lo que puede ocasionar errores y pérdida de información.

Por lo tanto, se identificó la necesidad de diseñar un programa que permita **registrar tareas, alumnos y calificaciones**, y posteriormente **mostrar todos los datos en pantalla**, de manera organizada y sin depender de almacenamiento externo.

Solución propuesta

La solución implementada fue un **programa en lenguaje C** con un **menú principal** que permite:

1. Gestionar las tareas.
2. Gestionar los alumnos.
3. Asignar calificaciones.
4. Mostrar todos los datos registrados.

El diseño se basa en el uso de **estructuras (struct)** para representar cada tipo de información:

- Tarea para guardar nombre y descripción.
- Alumno para registrar el nombre.
- calificación para asociar alumno, tarea y calificación.

Cada gestión se realiza mediante **funciones separadas**, favoreciendo la modularidad del código y la claridad en su mantenimiento.

El programa no utiliza archivos externos; toda la información se mantiene en memoria mientras está en ejecución. Esto permite un funcionamiento más rápido y adecuado para prácticas de laboratorio o ejercicios de aprendizaje.

Código:

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

#define MAX_ALUMNOS 50
#define MAX_TAREAS 50

// Estructuras principales

// Estructura para guardar los datos de una tarea
typedef struct {
    char nombre[50];
    char descripcion[100];
} Tarea;

// Estructura para guardar los datos de un alumno
typedef struct {
    char nombre[50];
} Alumno;

// Estructura que relaciona un alumno con una tarea y su calificación
typedef struct {
    int idAlumno;
    int idTarea;
    float calificacion;
} Calificacion;

// Variables globales
Tarea tareas[MAX_TAREAS];
Alumno alumnos[MAX_ALUMNOS];
Calificacion calificaciones[MAX_ALUMNOS * MAX_TAREAS];

int contadorTareas = 0;
int contadorAlumnos = 0;
int contadorCalificaciones = 0;

// Prototipos de funciones
```

```

void menu();
void gestionarTareas();
void gestionarAlumnos();
void asignarCalificaciones();
void mostrarDatos();

// -----
// Función principal
// -----
int main() {
    menu(); // Inicia el programa mostrando el menú principal
    return 0;
}

// Menú principal con opciones
void menu() {
    int opcion;
    do {
        printf("\n===== MENU PRINCIPAL =====\n");
        printf("1. Gestionar tareas\n");
        printf("2. Gestionar alumnos\n");
        printf("3. Asignar calificaciones\n");
        printf("4. Mostrar todos los datos\n");
        printf("0. Salir\n");
        printf("Seleccione una opcion: ");
        scanf("%d", &opcion);
        getchar(); // Limpia el buffer del teclado

        // Selección de opción según el número ingresado
        switch (opcion) {
            case 1: gestionarTareas(); break;
            case 2: gestionarAlumnos(); break;
            case 3: asignarCalificaciones(); break;
            case 4: mostrarDatos(); break;
            case 0: printf("Saliendo del programa...\n"); break;
            default: printf("Opcion no valida. Intente de nuevo.\n");
        }
    } while (opcion != 0);
}

// Función para registrar tareas
void gestionarTareas() {
    if (contadorTareas < MAX_TAREAS) {

```

```

printf("\n--- Registro de Tarea ---\n");

// Solicita y guarda el nombre de la tarea
printf("Nombre de la tarea: ");
fgets(tareas[contadorTareas].nombre, 50, stdin);
tareas[contadorTareas].nombre[strcspn(tareas[contadorTareas].nombre, "\n")] = 0;

// Solicita y guarda la descripción
printf("Descripcion: ");
fgets(tareas[contadorTareas].descripcion, 100, stdin);
tareas[contadorTareas].descripcion[strcspn(tareas[contadorTareas].descripcion, "\n")]
= 0;

    contadorTareas++; // Incrementa el contador
    printf("Tarea registrada exitosamente.\n");
} else {
    printf("No se pueden agregar más tareas.\n");
}
}

// Función para registrar alumnos

void gestionarAlumnos() {
    if (contadorAlumnos < MAX_ALUMNOS) {
        printf("\n--- Registro de Alumno ---\n");

        // Solicita el nombre del alumno
        printf("Nombre del alumno: ");
        fgets(alumnos[contadorAlumnos].nombre, 50, stdin);
        alumnos[contadorAlumnos].nombre[strcspn(alumnos[contadorAlumnos].nombre,
"\n")] = 0;

        contadorAlumnos++;
        printf("Alumno registrado exitosamente.\n");
    } else {
        printf("No se pueden agregar más alumnos.\n");
    }
}

// Función para asignar calificaciones

void asignarCalificaciones() {
    // Verifica si existen alumnos y tareas registrados
    if (contadorAlumnos == 0 || contadorTareas == 0) {
        printf("\nDebe haber al menos un alumno y una tarea registrados.\n");
        return;
    }
}

```

```

int i, j;
float calif;
printf("\n--- Asignacion de Calificaciones ---\n");

// Recorre todos los alumnos y tareas para asignar notas
for (i = 0; i < contadorAlumnos; i++) {
    for (j = 0; j < contadorTareas; j++) {
        printf("Calificacion de %s en la tarea '%s': ",
            alumnos[i].nombre, tareas[j].nombre);
        scanf("%f", &calif);
        getchar();

        // Guarda la calificación en el arreglo
        calificaciones[contadorCalificaciones].idAlumno = i;
        calificaciones[contadorCalificaciones].idTarea = j;
        calificaciones[contadorCalificaciones].calificacion = calif;
        contadorCalificaciones++;
    }
}
printf("Calificaciones asignadas correctamente.\n");
}

// Función para mostrar toda la información
void mostrarDatos() {
    printf("\n===== LISTADO DE TAREAS =====\n");
    for (int i = 0; i < contadorTareas; i++) {
        printf("%d. %s - %s\n", i + 1, tareas[i].nombre, tareas[i].descripcion);
    }

    printf("\n===== LISTADO DE ALUMNOS =====\n");
    for (int i = 0; i < contadorAlumnos; i++) {
        printf("%d. %s\n", i + 1, alumnos[i].nombre);
    }

    printf("\n===== CALIFICACIONES =====\n");
    // Muestra todas las calificaciones registradas
    for (int i = 0; i < contadorCalificaciones; i++) {
        int idA = calificaciones[i].idAlumno;
        int idT = calificaciones[i].idTarea;
        printf("%s - %s: %.2f\n",
            alumnos[idA].nombre, tareas[idT].nombre, calificaciones[i].calificacion);
    }
}

```

```
main.c [Practica 4] - Code::Blocks 25.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

===== LISTADO DE ALUMNOS =====
1. susana
2. carlos
3. marti
4. sofia
5. carla
6. marioano
7. javier
8. jose

===== CALIFICACIONES =====
susana - resolver un problema de matematicas: 8.00
susana - crear un programa: 6.00
carlos - resolver un problema de matematicas: 7.00
carlos - crear un programa: 5.00
marti - resolver un problema de matematicas: 9.00
marti - crear un programa: 10.00
sofia - resolver un problema de matematicas: 10.00
sofia - crear un programa: 8.00
carla - resolver un problema de matematicas: 7.00
carla - crear un programa: 6.00
marioano - resolver un problema de matematicas: 5.00
marioano - crear un programa: 0.00
javier - resolver un problema de matematicas: 10.00
javier - crear un programa: 2.00
jose - resolver un problema de matematicas: 8.00
jose - crear un programa: 10.00

===== MENU PRINCIPAL =====

C:\Users\LENOVO\OneDrive\Documents\Semestres\Tareas 2025-II\Intr.Ing.Software\... C/C++ Windows (CR+LF) WINDOWS-1252 Line 183, Col 1, Pos 5893 Insert Read/Write default
```

```
main.c [Practica 4] - Code::Blocks 25.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

--- Registro de Alumno ---
Nombre del alumno: marioano
Alumno registrado exitosamente.

===== MENU PRINCIPAL =====
1. Gestionar tareas
2. Gestionar alumnos
3. Asignar calificaciones
4. Mostrar todos los datos
0. Salir
Seleccione una opcion: 2

--- Registro de Alumno ---
Nombre del alumno: javier
Alumno registrado exitosamente.

===== MENU PRINCIPAL =====
1. Gestionar tareas
2. Gestionar alumnos
3. Asignar calificaciones
4. Mostrar todos los datos
0. Salir
Seleccione una opcion: 2

--- Registro de Alumno ---
Nombre del alumno: jose
Alumno registrado exitosamente.

===== MENU PRINCIPAL =====

C:\Users\LENOVO\OneDrive\Documents\Semestres\Tareas 2025-II\Intr.Ing.Software\... C/C++ Windows (CR+LF) WINDOWS-1252 Line 183, Col 1, Pos 5893 Insert Read/Write default
```

```
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
C:\Users\LENOVO\OneDrive\
4. Mostrar todos los datos
0. Salir
Seleccione una opcion: 3

--- Asignacion de Calificaciones ---
Calificacion de susana en la tarea 'resolver un problema de matematicas': 8
Calificacion de susana en la tarea 'crear un programa': 6
Calificacion de carlos en la tarea 'resolver un problema de matematicas': 7
Calificacion de carlos en la tarea 'crear un programa': 5
Calificacion de marti en la tarea 'resolver un problema de matematicas': 9
Calificacion de marti en la tarea 'crear un programa': 10
Calificacion de sofia en la tarea 'resolver un problema de matematicas': 10
Calificacion de sofia en la tarea 'crear un programa': 8
Calificacion de carla en la tarea 'resolver un problema de matematicas': 7
Calificacion de carla en la tarea 'crear un programa': 6
Calificacion de marioano en la tarea 'resolver un problema de matematicas': 5
Calificacion de marioano en la tarea 'crear un programa': 0
Calificacion de javier en la tarea 'resolver un problema de matematicas': 10
Calificacion de javier en la tarea 'crear un programa': 2
Calificacion de jose en la tarea 'resolver un problema de matematicas': 8
Calificacion de jose en la tarea 'crear un programa': 10
Calificaciones asignadas correctamente.

===== MENU PRINCIPAL =====
1. Gestionar tareas
2. Gestionar alumnos
3. Asignar calificaciones
4. Mostrar todos los datos
0. Salir
Seleccione una opcion: |
```

Tiempos de desarrollo

Etapas	Tiempo estimado	Tiempo real
Análisis del problema	20 min	15 min
Diseño del programa	25 min	30 min
Codificación	60 min	2 horas
Pruebas y corrección de errores	20 min	25 min
Total aproximado	2 h 05 min	3 h 10 min

Conclusiones

El desarrollo de este programa permitió aplicar los principios fundamentales de la **programación estructurada en C**, incluyendo el uso de **estructuras, arreglos, funciones y control mediante menús**.

La implementación demostró la importancia de organizar el código en módulos y trabajar con datos relacionados entre sí, reforzando la lógica para futuros proyectos con bases de datos o sistemas más complejos.

Aunque el programa no guarda información en archivos, cumple con los requisitos de gestión y visualización de datos, ofreciendo una base sólida para extenderlo posteriormente con persistencia de datos (por ejemplo, usando archivos .txt o bases de datos).