# Práctica No. 5

## Estructuras de datos.

**Objetivo:** Desarrollar aplicaciones de software utilizando estructuras de datos para modelar y manipular la información requerida para la solución de problemas, de forma analítica, propositiva y organizada..

**Material:**

- Computadora Personal (PC)
- Programa Editor de texto (ASCII), compilador GCC

**Equipo:**

- Computadora Personal

**Introducción**

ESTRUCTURAS (REGISTROS)

Una estructura es un tipo de dato compuesto que permite almacenar un conjunto de datos de diferente tipo. Los datos que contiene una estructura pueden ser de tipo simple (caracteres, números enteros o de punto flotante etc.) o a su vez de tipo compuesto (vectores, estructuras, listas, etc.).

A cada uno de los datos o elementos almacenados dentro de una estructura se les denomina miembros de esa estructura y éstos pertenecerán a un tipo de dato determinado. La definición de una estructura en C corresponde con la sintaxis siguiente:

```
struct miEstructura {
        tipo1 miembro1;
        tipo2 miembro2;
        ...
        tipoN miembroN;
};
```

Los miembros de una misma estructura deben tener nombres únicos, pero dos estructuras diferentes pueden contener miembros con el mismo nombre, sin problema. Toda definición de una estructura debe terminar con un punto y coma.

También es una práctica muy común asignarle un alias o sinónimo al nombre de la estructura, para evitar el tener que escribir "struct miEstructura" cada vez. C nos permite la posibilidad de hacer esto usando la palabra reservada typedef, lo que crea un alias a un tipo de dato ya definido:

typedef struct { ... } MiEstructura;
La estructura misma no tiene nombre (por la ausencia de nombre en la primera línea), pero tiene de alias "MiEstructura". Entonces se puede usar así:

```
MiEstructura variable;
```

Si la estructura es declarada antes de la declaración del alias de tipo entonces se puede usar:

```
typedef struct miEstructura MiEstructura;
```

y se puede usar de esta manera:

```
MiEstructura variable;
```

Note que es una convención, y una buena costumbre usar mayúscula en la primera letra de un sinónimo de tipo.

ESTRUCTURAS ANIDADAS

Una estructura puede estar dentro de otra estructura a esto se le conoce como anidamiento o estructuras anidadas.

**Teoría:**

Funciones de manipulación de cadenas de la biblioteca de manejo de cadenas.

**Desarrollo:**

El alumno deberá desarrollar un programa para resolver la siguiente problemática. Cada una de las funciones deberá estar en una **biblioteca** (ej. `Calificaciones.h`) implementada por el alumno (basarse en la práctica 2). El archivo que contenga el método main (ej. prac5.c) solo deberá desplegar el menú, declarar las variables y ejecutar la selección hecha por el usuario.

1) Implemente un programa para administrar las calificaciones de las prácticas de laboratorio del grupo 531, el cual puede visualizarse como se muestra en la **figura 1**, donde las columnas representan las prácticas y las filas corresponden a los alumnos. Las calificaciones y promedios deberán visualizarse en forma de tabla, tome las siguientes consideraciones.

    a) La tabla deberá ser flexible a modificación.
    b) Las calificaciones deberán ser generadas de forma aleatoria en un rango de 1 a 10.
    c) **promAx** corresponde al promedio obtenido por el alumno.
    d) **promPx** corresponde al promedio por práctica.

e) Haga uso de las estructuras que considere convenientes para manejar la información requerida.

| ID Práctica / ID Alumno | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 10 | 8 | 6 | 2 | ... | | | | | promA0 | A |
| 1 | | | | | | | | | | promA1 | N |
| ... | | | | | | | | | | promA2 | N |
| ... | | | | | | | | | | ... | ... |
| 19 | | | | | | | | ... | 7 | promA19 | A |
| | promP0 | promP1 | promP2 | …. | ... | ... | | | | | |

**Figura 1.** Calificaciones del grupo 531

2) Para ampliar las funcionalidades de nuestro sistema se pide manejar cierta información importante tanto del alumno como de la práctica por lo que se propone la definición de las estructuras de datos con los elementos que se describe en la **Tabla 1** y **Tabla 2**, seleccione el tipo de dato más adecuado a la información que debe ser almacenada.

| Nombre de la estructura de datos: **Alumno** | |
|---|---|
| **Elemento** | **Descripción** |
| `ID` | Número identificador del alumno |
| `nombre` | Contendrá el nombre del alumno |
| `correo` | Correo electrónico del alumno |
| `género` | Género del alumno |
| `ingreso` | Año en que ingresó a la universidad (formato "aaaa") |
| `periodo` | Periodo en que ingresó a la (periodos 1/ 2 en) |
| `promedio` | Promedio obtenido por el alumno |

**Tabla 1.** Elementos para la estructura de datos Alumno

| Nombre de la estructura de datos: **Practica** | |
|---|---|
| **Elemento** | **Descripción** |
| `ID` | Número identificador de la práctica |
| `titulo` | Título de la práctica |
| `comentario` | Comentario sobre la práctica |
| `promedio` | Promedio de todos los alumnos en la práctica |

**Tabla 2.** Elementos para la estructura de datos Practica

3) El número de fila(**ID Alumno** de la **Figura 1**) corresponde al número identificador del alumno (ID), el número de columna(**ID Práctica** de la **Figura 1**) corresponde al número identificador de la práctica(ID). Para poder tener esta relación se deben crear dos vectores, uno que contenga los registros de los alumnos y otro que contenga los registros de las prácticas.

4) Calcule la media de las calificaciones obtenidas por los alumnos.

5) Implemente una forma de saber si el alumno ha aprobado el curso o no, siendo **A**: aprobado y **N**: No aprobado.

6) El programa debe mostrar un menú al usuario de las acciones que puede realizar.
   a) Mostrar tabla de calificaciones
   b) Mostrar toda la información de un alumno indicado por el usuario (ID).
   c) Mostrar toda la información de una práctica indicada por el usuario (ID).
   d) Mostrar lista de alumnos (aplicar diseño).
   e) Mostrar lista de prácticas (aplicar diseño).
   f) Mostrar al alumno (o alumnos) con el más alto promedio.
   g) Mostrar a todos los alumnos que tienen calificación reprobatoria.
   h) Mostrar las prácticas con promedio menor a 6.
   i) Salir

**Restricciones**

1. El código fuente deberá contener comentarios que indique el objetivo del programa y descripción de instrucciones más relevantes.
2. Evitar nombres de variables y funciones que no reflejen su propósito claramente.
3. Se debe evitar repetir código en medida de lo posible, se deben realizar las funciones de tal forma que puedan ser reutilizables por otras funciones.

**Conclusiones y Comentarios.**