Universidad Autónoma de Baja California
FACULTAD DE CIENCIAS QUÍMICAS E INGENIERÍA

Reporte de tarea.

Lenguaje C (531).

Tarea No. 1:

Uso de apuntadores para estructuras con arreglos.

Alumno

Joshua Osorio Osorio - 1293271

Docente

Fernando Elihonai Saucedo Lares

5 de mayo de 2025

ÍNDICE

Objetivo

Evaluar el manejo de estructuras de datos y el uso de funciones con parámetros para organizar el código de manera eficiente.

Introducción

Declaración de Apuntadores para Estructuras con Arreglos.

Los apuntadores a estructuras con arreglos permiten manipular datos complejos de manera dinámica y eficiente. Se declaran combinando la sintaxis de apuntadores, estructuras y arreglos:

```C/C++
typedef struct {
    int id;
    char nombre[50];
} Estudiante;

// Declaración de un arreglo de estructuras
Estudiante lista_estudiantes[10];
```

```
// Declaración de un apuntador a un arreglo de estructuras
Estudiante *ptr_estudiantes = lista_estudiantes;
```

Sintaxis para Acceso y Manipulación.

Acceso a miembros de estructuras con arreglos:

Notación de corchetes []:

```
C/C++
ptr_estudiantes[0].id = 100; // Accede al primer estudiante
strcpy(ptr_estudiantes[0].nombre, "Juan");
```

Notación de apuntador con operador ->:

```
C/C++
(ptr_estudiantes + 1)->id = 101; // Accede al segundo estudiante
strcpy((ptr_estudiantes + 1)->nombre, "Ana");
```

Asignación dinámica de memoria:

```
C/C++
Estudiante *ptr_dinamico = (Estudiante*)malloc(10 * sizeof(Estudiante));
ptr_dinamico[0].id = 200; // Uso como arreglo
free(ptr_dinamico); // Liberar memoria
```

Indicadores importantes:

- Operador ->: Usado exclusivamente con apuntadores para acceder a miembros de estructuras.
- Aritmética de apuntadores: Permite navegar entre elementos del arreglo (ptr + i).

Código de ejemplo <main.c>:

```C/C++
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct {
    int matricula;
    char nombre[50];
    float promedio;
} Estudiante;

int main() {
    // 1. Declaración de un arreglo estático de estructuras
    Estudiante grupo[3] = {
        {101, "Ana López", 8.5},
        {102, "Carlos Ruiz", 9.1},
        {103, "María García", 7.8}
    };

    // 2. Declaración de un apuntador a estructuras
    Estudiante *ptr_estudiante = grupo; // Apunta al primer elemento del
arreglo

    // 3. Acceso a los datos usando notación de apuntador
    printf("\n--- Acceso con notación de apuntador (->) ---\n");
    for (int i = 0; i < 3; i++) {
        printf("Estudiante %d:\n", i + 1);
        printf("  Matrícula: %d\n", (ptr_estudiante + i)->matricula);
        printf("  Nombre: %s\n", (ptr_estudiante + i)->nombre);
        printf("  Promedio: %.2f\n\n", (ptr_estudiante + i)->promedio);
    }

    // 4. Modificación de datos usando aritmética de apuntadores
    (ptr_estudiante + 1)->promedio = 9.5; // Actualiza el promedio de Carlos

    // 5. Asignación dinámica de memoria
    Estudiante *ptr_dinamico = (Estudiante*)malloc(2 * sizeof(Estudiante));
    ptr_dinamico[0].matricula = 104;
```

```c
    strcpy(ptr_dinamico[0].nombre, "Luisa Méndez");
    ptr_dinamico[0].promedio = 8.9;

    ptr_dinamico[1].matricula = 105;
    strcpy(ptr_dinamico[1].nombre, "Jorge Silva");
    ptr_dinamico[1].promedio = 9.2;

    printf("\n--- Datos en memoria dinámica (notación de arreglo []) ---\n");
    for (int i = 0; i < 2; i++) {
        printf("Estudiante %d:\n", i + 4);
        printf("  Matrícula: %d\n", ptr_dinamico[i].matricula);
        printf("  Nombre: %s\n", ptr_dinamico[i].nombre);
        printf("  Promedio: %.2f\n\n", ptr_dinamico[i].promedio);
    }

    // Liberar memoria
    free(ptr_dinamico);

    return 0;
}
```

Conclusiones

El uso de apuntadores a estructuras que contienen arreglos permite un manejo más flexible y eficiente de los datos, especialmente cuando se requiere asignación dinámica de memoria. La notación -> es fundamental para acceder a los miembros de una estructura mediante un apuntador, mientras que en arreglos estáticos se utiliza la notación punto (.). La combinación de estas herramientas optimiza significativamente el acceso y la manipulación de datos complejos, resultando especialmente útil en aplicaciones como bases de datos, sistemas de inventario o cualquier sistema que requiera organización estructurada de grandes volúmenes de información.

Bibliografía

[1] “IBM i,” 08-Apr-2025. [Online]. Available: https://www.ibm.com/docs/es/i/7.5.0?topic=functions-snprintf-print-formatted-data-buffer.

[2] “Asignar valor a cadena de caracteres en estructura,” Stack Overflow En Español. [Online]. Available: https://es.stackoverflow.com/questions/570958/asignar-valor-a-cadena-de-caracteres-en-estructura.

[3] TylerMSFT, “Declaraciones de puntero,” Microsoft Learn. [Online]. Available: https://learn.microsoft.com/es-es/cpp/c-language/pointer-declarations?view=msvc-170.