

# PROGRAMACIÓN

**Unidad 7 – Parte 2.** Estructuras y funciones. Arreglos de estructuras.

# Revisamos la tarea pendiente



## 1. ¿Qué es una estructura en C y para qué se utiliza?

*Una estructura en C es un tipo de dato que permite combinar varias variables bajo un único nombre. Se utiliza para representar registros de datos que contienen múltiples atributos relacionados.*

## 2. ¿Cuál es la diferencia entre una estructura y un arreglo en C?

*Un arreglo en C almacena elementos del mismo tipo de datos en una secuencia contigua, mientras que una estructura puede contener varios campos de diferentes tipos de datos relacionados.*

## 3. ¿Cuál es la diferencia entre una estructura y una unión en C?

*A diferencia de una estructura, una unión en C almacena todos sus campos en la misma ubicación de memoria, lo que significa que solo puede contener un valor a la vez. En una estructura, cada campo tiene su propia ubicación de memoria.*

## 4. ¿Cómo se declara una estructura en C?

*Una estructura se declara utilizando la palabra clave `struct`, seguida del nombre de la estructura y una lista de miembros entre llaves.*

## 5. ¿Cómo se accede a los miembros de una estructura en C?

*Los miembros de una estructura se acceden utilizando el operador de punto (`.`). Por ejemplo: `persona.edad`*

# Revisamos la tarea pendiente



6. ¿Qué es una estructura anidada y cómo se utiliza?

*Una estructura anidada es una estructura que se declara dentro de otra estructura. Se utiliza para representar relaciones entre objetos más complejas. Puedes acceder a los miembros de estructuras anidadas utilizando el operador de punto. Por ejemplo: persona.fecha.anio*


7. ¿Cómo se inicializa una estructura en C?

*Una estructura se puede inicializar proporcionando valores para sus miembros durante la declaración o utilizando la notación de inicialización por campo*

8. ¿Cómo crearías una estructura para representar un punto en un espacio tridimensional?

```
struct Punto3D {  
    float x;  
    float y;  
    float z;  
};
```

# Usar campos de una estructura como argumentos de funciones



- ❑ Se trabajara con el campo de la estructura usando el operador punto.
- ❑ Cuando se pasa un campo o miembro de una estructura a una función como argumento, en realidad se pasa el valor de ese campo.

# OPERACIONES



Usar variables estructuradas por partes como argumentos de funciones:

- Al pasar un campo de la estructura a una función, se puede usar el mecanismo de pasaje por valor o por referencia.

# Ejemplo

Supongamos que tenemos una estructura de nombre **Alumno** que almacena los datos de un alumno como ser sus datos personales (apellido, nombre, edad) y las notas trimestrales del mismo. Nuestra tarea será realizar una función que calcule el promedio de 3 notas.

```
struct{  
    char nombre[100];  
    char apellido[100];  
    int edad;  
    float nota1;  
    float nota2;  
    float nota3;  
}typedef Alumno;
```

```
float promedio(float nota1, float nota2, float nota3);  
int main()  
{  
    Alumno estudiante;  
    float prom;  
  
    printf("Ingrese el Apellido del alumno: ");  
    gets(estudiante.apellido);  
  
    printf("\nIngrese el Nombre del alumno: ");  
    gets(estudiante.nombre);  
  
    printf("\nIngrese la edad del alumno: ");  
    scanf("%d", &estudiante.edad);  
  
    printf("\nIngrese la 1ra nota:");  
    scanf("%f", &estudiante.nota1);  
  
    printf("\nIngrese la 2da nota:");  
    scanf("%f", &estudiante.nota2);  
  
    printf("\nIngrese la 3ra nota:");  
    scanf("%f", &estudiante.nota3);  
  
    prom = promedio(estudiante.nota1, estudiante.nota2, estudiante.nota3);  
  
    printf("\nEl promedio del almno es: %f", prom);  
  
    return 0;  
}  
  
float promedio(float nota1, float nota2, float nota3)  
{  
    return ((nota1 + nota2 + nota3) / 3);  
}
```

# Ejemplo

Podemos modificar la nota del alumno según la consideración que se le de en la materia.

```
struct{  
  
    char nombre[100];  
    char apellido[100];  
    int edad;  
    float nota1;  
    float nota2;  
    float nota3;  
  
}typedef Alumno;
```

```
void cambiarNotas(float *pnota1, float *pnota2, float *pnota3);  
int main()  
{  
    Alumno estudiante;  
    float prom;  
  
    printf("Ingrese el Apellido del alumno: ");  
    gets(estudiante.apellido);  
  
    printf("\nIngrese el Nombre del alumno: ");  
    gets(estudiante.nombre);  
  
    printf("\nIngrese la edad del alumno: ");  
    scanf("%d", &estudiante.edad);  
  
    printf("\nIngrese la 1ra nota:");  
    scanf("%f", &estudiante.nota1);  
  
    printf("\nIngrese la 2da nota:");  
    scanf("%f", &estudiante.nota2);  
  
    printf("\nIngrese la 3ra nota:");  
    scanf("%f", &estudiante.nota3);  
  
    cambiarNotas(&estudiante.nota1, &estudiante.nota2, &estudiante.nota3);  
    prom = promedio(estudiante.nota1, estudiante.nota2, estudiante.nota3);  
  
    printf("\nEl promedio del alumno es: %f", prom);  
  
    return 0;  
}  
  
void cambiarNotas(float *pnota1, float *pnota2, float *pnota3)  
{  
    *pnota1 = (*pnota1) * 0.2;  
    *pnota2 = (*pnota2) * 0.2;  
    *pnota3 = (*pnota3) * 0.6;  
}
```

# OPERACIONES

Usar variables estructuradas completas como argumentos de funciones:

- ❑ Al pasar una estructura completa a una función, se usa el mecanismo de pasaje por valor.
- ❑ Sabemos que de esta manera cualquier cambio en los contenidos de los campos de la estructura dentro de la función, no se reflejan en el bloque que la invoco.



# Ejemplo

Continuemos con el ejemplo anterior

```
struct{  
    char nombre[100];  
    char apellido[100];  
    int edad;  
    float nota1;  
    float nota2;  
    float nota3;  
}typedef Alumno;
```

```
void mostrarDatos(Alumno alum)  
{  
    puts("Los datos cargado del alumno es");  
    printf("El Apellido del alumno es: ");  
    puts(alum.apellido);  
  
    printf("\nEl Nombre del alumno es: ");  
    puts(alum.nombre);  
  
    printf("\nLa edad del alumno es: %d", alum.edad);  
  
    printf("\nLa 1ra nota es: %d", alum.nota1);  
  
    printf("\nLa 2da nota es: %d", alum.nota2);  
  
    printf("\nLa 3ra nota es: %d", alum.nota3);  
}
```

```
void mostrarDatos(Alumno alum);  
int main()  
{  
    Alumno estudiante;  
    float prom;  
  
    printf("Ingrese el Apellido del alumno: ");  
    gets(estudiante.apellido);  
  
    printf("\nIngrese el Nombre del alumno: ");  
    gets(estudiante.nombre);  
  
    printf("\nIngrese la edad del alumno: ");  
    scanf("%d", &estudiante.edad);  
  
    printf("\nIngrese la 1ra nota:");  
    scanf("%f", &estudiante.nota1);  
  
    printf("\nIngrese la 2da nota:");  
    scanf("%f", &estudiante.nota2);  
  
    printf("\nIngrese la 3ra nota:");  
    scanf("%f", &estudiante.nota3);  
  
    mostrarDatos(estudiante);  
    return 0;  
}
```

# Ejemplo

Mejoremos el ejemplo anterior

```
struct{  
    char nombre[100];  
    char apellido[100];  
    int edad;  
    float nota1;  
    float nota2;  
    float nota3;  
}typedef Alumno;
```

```
void mostrarDatos(Alumno alum)  
{  
    puts("Los datos cargado del alumno es");  
    printf("El Apellido del alumno es: ");  
    puts(alum.apellido);  
  
    printf("\nEl Nombre del alumno es: ");  
    puts(alum.nombre);  
  
    printf("\nLa edad del alumno es: %d", alum.edad);  
  
    printf("\nLa 1ra nota es: %d", alum.nota1);  
  
    printf("\nLa 2da nota es: %d", alum.nota2);  
  
    printf("\nLa 3ra nota es: %d", alum.nota3);  
}
```

```
Alumno cargarDatos();  
int main()  
{  
    Alumno estudiante;  
  
    estudiante = cargarDatos();  
  
    mostrarDatos(estudiante);  
    return 0;  
}
```

```
Alumno cargarDatos()  
{  
    Alumno alum;  
  
    printf("Ingrese el Apellido del alumno: ");  
    gets(alum.apellido);  
  
    printf("\nIngrese el Nombre del alumno: ");  
    gets(alum.nombre);  
  
    printf("\nIngrese la edad del alumno: ");  
    scanf("%d", &alum.edad);  
  
    printf("\nIngrese la 1ra nota:");  
    scanf("%f", &alum.nota1);  
  
    printf("\nIngrese la 2da nota:");  
    scanf("%f", &alum.nota2);  
  
    printf("\nIngrese la 3ra nota:");  
    scanf("%f", &alum.nota3);  
  
    return (alum);  
}
```

# Arreglo de estructura

- Ahora necesitamos sacar el promedio de notas de un grupo de 20 alumnos.
- ¿Qué deberíamos hacer para responder a ese requisito?
- Tengo que declarar 20 variables de tipo struct?

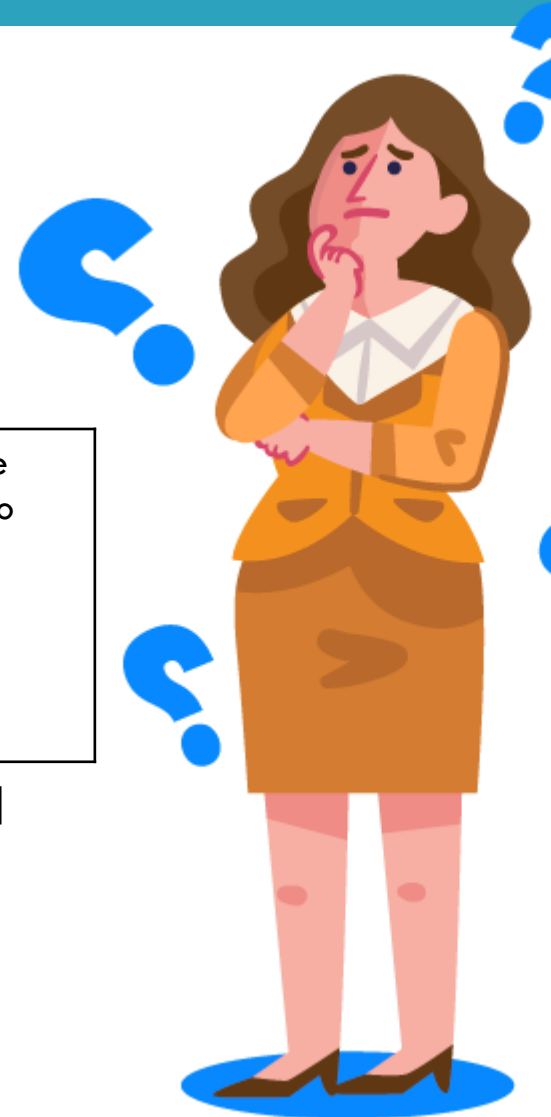


# Arreglo de estructura

- ¿Cómo sería un arreglo con todos los datos de un alumno?

Alumnos

Nombre	Nombre	Nombre		Nombre
Apellido	Apellido	Apellido		Apellido
Edad	Edad	Edad		Edad
Nota1	Nota1	Nota1		Nota1
Nota2	Nota2	Nota2		Nota2
Nota3	Nota3	Nota3		Nota3
[0]	[1]	[2]	...	[n]



# Arreglo de estructura

- Respondemos esas preguntas desarrollando el siguiente ejemplo:

```
struct Alumno{  
    char nom[100];  
    char ape[100];  
    int edad;  
    float nota1;  
    float nota2;  
    float nota3;  
};
```

En este ejemplo podemos ver que trabajamos con arreglos de estructuras, haciendo uso del índice, como lo veníamos haciendo con los arreglos en general.

```
float promedio(float nota1, float nota2, float nota3);  
  
int main()  
{  
    struct Alumno estudiante[20];  
    float prom;  
  
    for (int i = 0; i < 20; i++)  
    {  
        printf("Datos del alumno N°: %d", i);  
  
        puts("Ingrese el apellido del alumno : ");  
        gets(estudiante[i].ape);  
  
        puts("Ingrese el nombre del alumno:");  
        gets(estudiante[i].nom);  
  
        puts("Ingrese la edad del alum:");  
        scanf("%d", &estudiante[i].edad);  
  
        puts("Ingrese 1ra nota:");  
        scanf("%f", &estudiante[i].nota1);  
  
        puts("Ingrese 2da nota:");  
        scanf("%f", &estudiante[i].nota2);  
  
        puts("Ingrese 3ra nota:");  
        scanf("%f", &estudiante[i].nota3);  
  
        prom = promedio(estudiante[i].nota1, estudiante[i].nota2, estudiante[i].nota3);  
  
        printf("El promedio del alumno es: %f", prom);  
    }  
  
    return 0;  
}  
  
float promedio(float nota1, float nota2, float nota3)  
{  
    return((nota1+nota2+nota3)/3);  
}
```

# Arreglo de estructura

¿Como inicializar un arreglo de estructura?

```
#include <stdio.h>

struct{
    char nombre[50];
    int edad;
}typedef Persona;

int main() {
    // Inicialización de un arreglo de estructuras
    Persona clientes[3] = {
        {"Juan", 25},
        {"Maria", 30},
        {"Pedro", 22}
    };

    for (int i = 0; i < 3; i++) {
        printf("Persona %d:\n", i + 1);
        printf("Nombre: %s\n", clientes[i].nombre);
        printf("Edad: %d\n", clientes[i].edad);
        printf("\n");
    }

    return 0;
}
```

# Tarea para la casa



1. Explica cómo puedes pasar una estructura como argumento a una función en C.
2. ¿Puedes devolver una estructura desde una función en C? ¿Cómo?
3. Proporciona un ejemplo de cómo usar una estructura en una función.
4. ¿Cómo se declara un arreglo de estructuras en C?
5. Explica cómo se inicializan los elementos de un arreglo de estructuras.
6. ¿Cuál es la diferencia entre acceder a un miembro de una estructura en una estructura individual y en un arreglo de estructuras?
7. ¿Cómo se pasa un arreglo de estructuras a una función en C?

**Lo revisamos la próxima clase.**

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

```
int main()  
{
```

```
    printf("Hasta la próxima clase!!\n");
```

```
    return 0;
```

```
}
```