

# Estructuras

## Programación Digital I

**Gilberto Diaz**  
**gilberto@ula.ve**  
**Universidad de Los Andes**  
**Facultad de Ingeniería**  
**Escuela de Sistemas**  
**Depto de Computación**  
**Mérida - Venezuela**

# Definición

Una estructura o registro es una colección de una o más variables que pueden ser manipuladas como una sola. Este conjunto puede estar compuesto de variables de cualquier tipo.

Las estructuras también son conocidas como registros.

Podemos ver a las estructuras como nuevos tipos de datos (datos definidos por el usuario)

Un ejemplo:

Estructura RegistroEstudiante

```
{  
  cadena cedula;  
  cadena nombre;  
  cadena telefono;  
  entero notaPrimerParcial;  
  entero notaSegundoParcial;  
};
```

## **Ventajas:**

- Mejor organización del código, más expresividad y en consecuencia un mejor código.
- Acorta el tamaño del código para hacer asignaciones de estructuras completas.

Ej: `estudiante1 = estudiante2`. Donde ambas variables son del tipo `RegistroEstudiante`.

- Pueden ser pasadas por parámetro a las funciones y a su vez pueden ser devueltas como la salida de la función

# Notación Algorítmica:

Estructura NombreDeEstructura

```
{  
    tipo1 dato1;  
    tipo2 dato2;  
    .  
    .  
    .  
    tipon daton;  
}
```

## Notación en C:

```
struct point  
{  
    int coordx;  
    int coordy;  
};
```

## **Propiedad Interesante:**

Las estructuras pueden anidarse, es decir, uno de los datos puede ser otra estructura.

Podemos declarar una o mas variables del tipo estructura de la siguiente manera:

```
struct
{
    int var1;
    float var2;
} i,j,k;
```

Una vez definida la estructura, pueden hacerse sucesivas declaraciones:

```
struct point    w;
```

Y también puede utilizarse la inicialización con llaves:

```
struct point h = {10, 20};
```

# Operaciones con Estructuras

Para trabajar con estructuras hace falta conocer sus operadores.

Operador de acceso a los miembros:

```
nombre_estructura.miembro = valor;
```

Podemos hacer arreglos de estructuras

```
struct point puntos[3];
```

Para acceder los elementos:

```
puntos[0].coordx = 1;
```

## Ejemplo

```
struct point
{
    int x;
    int y;
};
struct triangle {
    struct point p1, p2, p3;
} t;
int main ()
{
    t.p1 = {4, 6};
    t.p2.x = 20;
    t.p2.y = 32;
    t.p3.x = t.p2.x - 10;
    t.p3.y = t.p3.y + t.p3.x;
return 0;
}
```

## Ejemplo

```
typedef struct point{  
    float coordx;  
    float coordy;  
};
```

```
int main(){  
    point x,y,z;  
    .  
    .  
    .
```