

SESION 10

Algorítmica I



Estructura de Regis

Registros y Estructuras

Tipos de Datos Definidos por el Usuario

ESTRUCTURAS

- 1. Concepto de estructura**
- 2. Definición del tipo de dato estructura**
- 3. Declaración de variables de tipo estructura**
- 4. Inicialización de variables de tipo estructura**
- 5. Acceso a los miembros de una estructura**
- 6. Uso de estructuras en asignaciones**
- 7. Estructuras anidadas**
- 8. Arrays de estructuras**

TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Introducción

Hemos visto anteriormente el tipo de dato compuesto **ARRAY** definido como una colección de elementos del mismo tipo.

Nombres =

"Ana"	"Pedro"	"Antonio"	...	"Luis"
0	1	2		N-1

Elementos
de tipo cadena

Teléfonos =

6129215	6154215	6144258	...	6165024
---------	---------	---------	-----	---------

Elementos
de tipo entero

En algunos casos nos puede interesar trabajar con colecciones de elementos de distinto tipo:

Alumno =

"Ana"	6129215	1	"Sevilla"
-------	---------	---	-----------

Nombre
(tipo cadena)

Teléfono
(tipo int)

Curso
(tipo int)

Lugar de nacimiento
(tipo cadena)

ESTRUCTURA

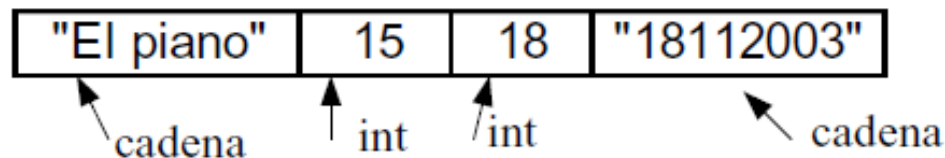
TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Concepto de estructura:

- Una **estructura** es una colección de uno o más elementos, cada uno de los cuales puede ser de un tipo de dato diferente.
- Cada elemento de la estructura se denomina **miembro**.
- Una estructura puede contener un número ilimitado de miembros.
- A las estructuras también se las llama **registros**.

Ejemplo:

Podemos crear una estructura llamada **disco** que contiene 4 miembros: **título del disco**, **número de canciones**, **precio** y **fecha de compra**.



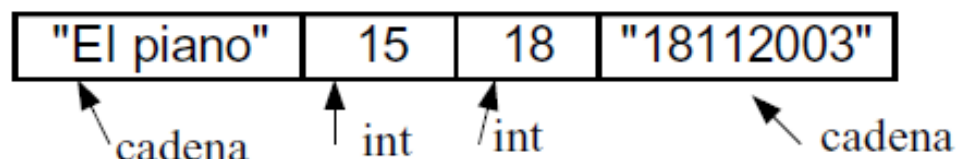
TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Concepto de estructura:

- Una **estructura** es una colección de uno o más elementos, cada uno de los cuales puede ser de un tipo de dato diferente.
- Cada elemento de la estructura se denomina **miembro**.
- Una estructura puede contener un número ilimitado de miembros.
- A las estructuras también se las llama **registros**.

Ejemplo:

Podemos crear una estructura llamada **disco** que contiene 4 miembros: **título del disco**, **número de canciones**, **precio** y **fecha de compra**.



TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Ejemplos:

Para definir el tipo de dato **disco** como una estructura con 4 miembros:

```
void main( )
{
    ....
    struct disco
    {
        char titulo[30];
        int num_canciones;
        float precio;
        char fecha_compra[8];
    };

    ...
}
```

Declaración del tipo de dato **complejo** como una estructura con 2 miembros.

```
void main( )
{
    ....
    struct complejo
    {
        int real;
        int imaginaria;
    };

    ...
}
```

Declaración de variables de tipo estructura:

Una vez definido el tipo de dato estructura, necesitamos declarar variables de ese tipo (como para cualquier tipo de dato !!!).

Existen dos formas diferentes:

➡ En la definición del tipo de datos estructura.

```
struct complejo  
{  
    int real;  
    int imaginaria;  
} comp1, comp2, comp3 ;
```

➡ Como el resto de las variables.

```
...  
complejo comp4, comp5 ;  
...
```

- ◆ Los miembros de cada variable se almacenan en posiciones consecutivas en memoria.
- ◆ El compilador reserva la memoria necesaria para almacenar las 5 variables.

Inicialización de variables de tipo estructura:

Las variables de tipo estructura las podemos inicializar de dos formas:

1. Inicialización en el cuerpo del programa: Lo veremos más adelante.

2. Inicialización en la declaración de la variable:

Se especifican los valores de cada uno de los miembros entre llaves y separados por comas.

```
struct complejo
{
    int real;
    int imaginaria;
} comp1= {25, 2} ;
```

```
...
complejo comp4 = {25, 2} ;
...
```


TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Inicialización de variables de tipo estructura:

```
void main( )  
{  
    ....  
    struct disco  
    {  
        char titulo[30];  
        int num_canciones;  
        float precio;  
        char fecha_compra[8];  
    };  
    .....  
    disco cd = { "El piano", 15, 18, "18112003" };  
    ...  
}
```

Mas ejemplos de inicialización
de variables de tipo **disco**

Definición de la estructura
(formato)

¿Cuánta memoria reserva
el compilador ?

cd =

"El piano"

15

18

"18112003"

TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Acceso a los miembros de una variable de tipo estructura:

Una vez que hemos declarado una variable de tipo estructura, podemos acceder a los miembros de dicha variable:

cd =

"El piano"	15	18	"18112003"
------------	----	----	------------

Nos puede interesar *modificar* la información de alguno de los miembros, *recuperar* información para imprimirla por pantalla, etc.

El acceso a los miembros se puede hacer de dos formas:

Utilizando el operador punto (.)

Selector
directo

variable.miembro

cd.titulo , cd.precio , cd.num_canciones

Utilizando el operador puntero (->)

Selector
indirecto

variable -> miembro

Lo veremos
más
adelante


Acceso a los miembros de una variable de tipo estructura:

```
struct disco
{
    char titulo[30];
    int num_canciones;
    float precio;
    char fecha_compra[8];
};

....
disco cd;

...
cd.titulo = "El piano";
cd.num_canciones = 15;
cd.precio = 18;
cd.fecha_compra = "18112003";
...
```

Inicialización de la
variable cd1



cd =

"El piano"	15	18	"18112003"
------------	----	----	------------

TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Acceso a los miembros de una variable de tipo estructura:

```
struct disco
{
    char titulo[30];
    int num_canciones;
    float precio;
    char fecha_compra[8];
};
```

```
.....
disco cd;
```

```
...
cout << "\n Introduzca título" ;
cin.getline ( cd.titulo, 30 ) ;
cout << "\n Introduzca precio" ;
cin>> cd.precio ;
```

```
....
cout << cd.titulo;
precio_final = cd.precio - 10;
cd.precio = precio_final;
```

Almacenar información en la
variable cd
mediante el teclado

Recuperar y modificar información
de la variable cd

Uso de variables de tipo estructura en asignaciones:

Se puede asignar una estructura a otra de la siguiente manera:

```
struct disco
{
    char titulo[30];
    int num_canciones;
    float precio;
    char fecha_compra[8];
};
.....
disco cd1 = { "El piano", 15, 18, "18112003" };
....
disco cd2, cd3;

cd2 = cd1;
cd2 = cd3 = cd1;
```

TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Estructuras anidadas:

Se pueden definir estructuras donde uno o varios de los miembros son a su vez de tipo estructura.

struct direccion

```
{  
    char calle[30];  
    int numero;  
    int cod_postal ;  
    char poblacion[30];  
};
```

struct telefonos

```
{  
    char casa[13] ={'\0'};  
    char oficina[13] ={'\0'};  
    char movil[13] ={'\0'};  
};
```

Ejemplo

struct cliente

```
{  
    char nombre[30];  
    direccion dir;  
    telefonos tlf ;  
    int edad;  
};
```

cliente c;

```
c.nombre = "Ana Gonzalez";  
c.dir.calle = "Av. Europa";  
c.dir.numero = 12;  
c.tlf.movil = "602.23.23.23";  
c.tlf.casa = "91.123.45.67";
```

El acceso a los miembros requiere el uso múltiple del operador punto (.)

TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Estructuras anidadas:

Ejemplo

```
struct direccion
{
    char calle[30];
    int numero;
    int cod_postal ;
    char poblacion[30];
};
```

```
struct telefonos
{
    char casa[12] ={'\0'};
    char oficina[12]; ={'\0'};
    char movil[12] ={'\0'};
};
```

```
struct cliente
{
    char nombre[30];
    direccion casa;
    direccion oficina ;
    telefonos tlf;
    int edad;
} ;
```

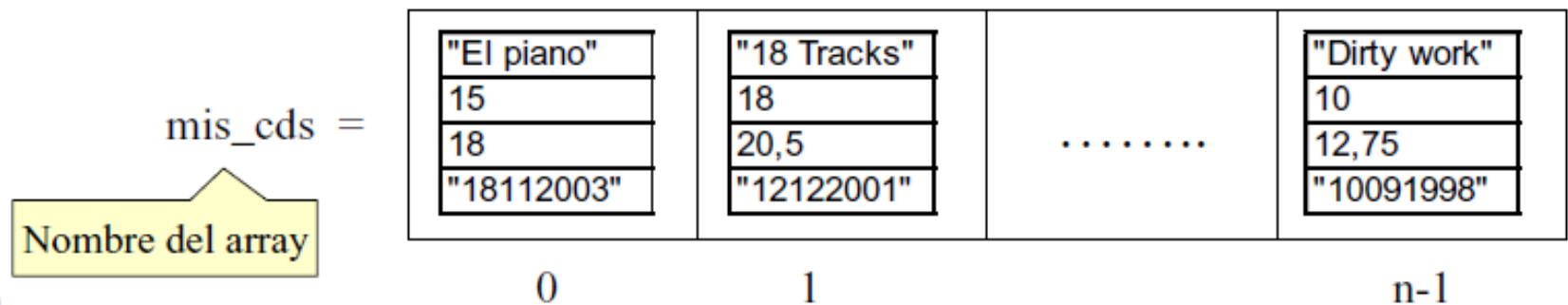
cliente c;

```
...
c.nombre = "Ana Gonzalez";
c.casa.calle = "Av. Europa";
c.casa.numero = 45;
c.oficina.calle = "Pirineos";
c.tlf.oficina ="602.23.23.23";
c.tlf.casa = "91.123.45.67";
```


TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Arrays de estructuras:

Supongamos que queremos guardar información de todos los discos que tenemos en nuestra casa. Con una variable de tipo disco, solo podemos guardar los datos de uno. Necesitaremos un array de discos:

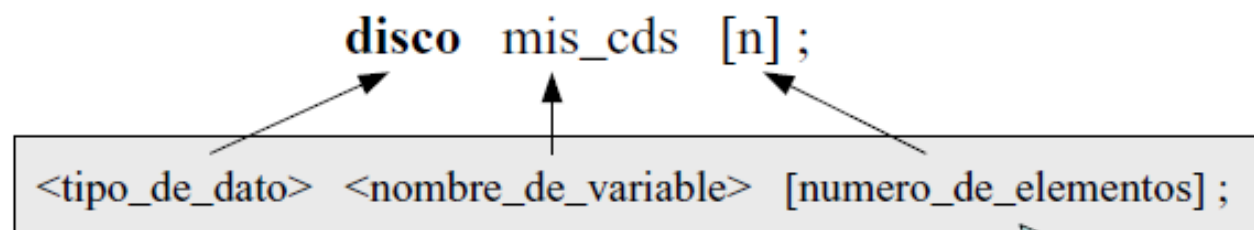


- ➡ Tenemos una colección de elementos del mismo tipo.
- ➡ Se utilizan mucho los arrays de estructuras como método para almacenar datos en un archivo y leer datos de un archivo.

TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Arrays de estructuras. Declaración:

La declaración de un array de estructuras es igual a cualquier otro array:



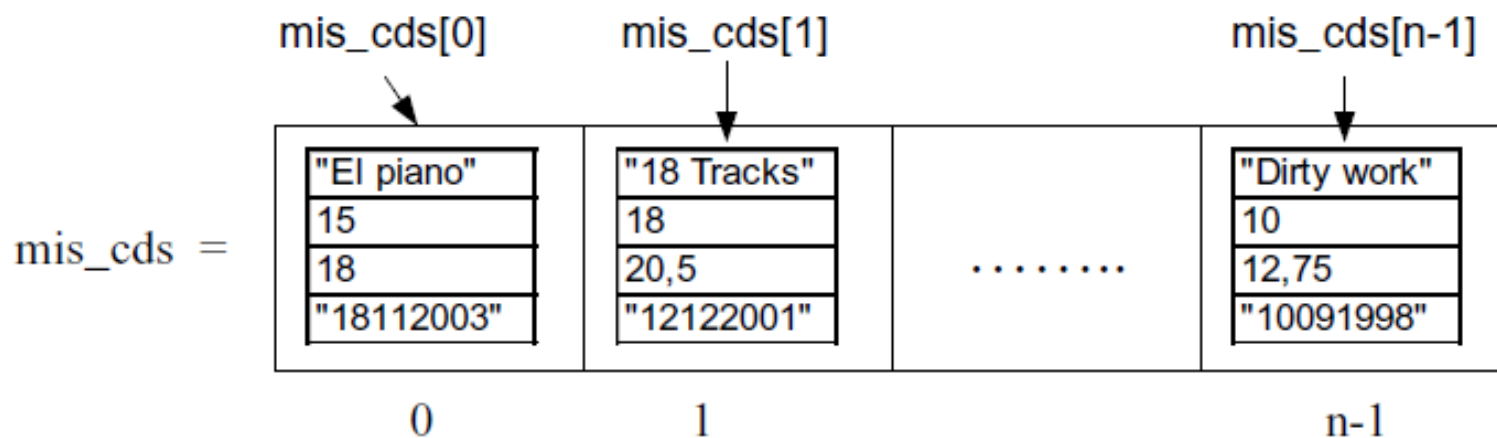
Sintaxis de la declaración
de un array

mis_cds =

<table><tr><td>"El piano"</td></tr><tr><td>15</td></tr><tr><td>18</td></tr><tr><td>"18112003"</td></tr></table>	"El piano"	15	18	"18112003"	<table><tr><td>"18 Tracks"</td></tr><tr><td>18</td></tr><tr><td>20,5</td></tr><tr><td>"12122001"</td></tr></table>	"18 Tracks"	18	20,5	"12122001"	<table><tr><td>"Dirty work"</td></tr><tr><td>10</td></tr><tr><td>12,75</td></tr><tr><td>"10091998"</td></tr></table>	"Dirty work"	10	12,75	"10091998"
"El piano"															
15															
18															
"18112003"															
"18 Tracks"															
18															
20,5															
"12122001"															
"Dirty work"															
10															
12,75															
"10091998"															
0	1		n-1												

Arrays de estructuras. Acceso a los elementos:

El acceso a los elementos se hace como en un array normal, se pone el nombre del array seguido del índice del elemento al que queremos acceder.



Ahora, si queremos acceder a los miembros de los elementos del array:

```
mis_cds[0].titulo = "El piano" ;  
mis_cds[1].precio = 20,5 ;
```

```
cin>> mis_cds[0].num_canciones ;  
cout << mis_cds[1].titulo ;
```

Estructuras

Parte II

TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Estructuras anidadas:

Se pueden definir estructuras donde uno o varios de los miembros son a su vez de tipo estructura.

```
struct direccion
{
    char calle[30];
    int numero;
    int cod_postal ;
    char poblacion[30];
};
```

```
struct telefonos
{
    char casa[13] ={'\0'};
    char oficina[13] ={'\0'};
    char movil[13] ={'\0'};
};
```

Ejemplo

```
struct cliente
{
    char nombre[30];
    direccion dir;
    telefonos tlf ;
    int edad;
};
```

```
cliente c;
c.nombre = "Ana Gonzalez";
c.dir.calle = "Av. Europa";
c.dir.numero = 12;
c.tlf.movil = "602.23.23.23";
c.tlf.casa = "91.123.45.67";
```

El acceso a los miembros requiere el uso múltiple del operador punto (.)

TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Estructuras anidadas:

Ejemplo

```
struct direccion
{
    char calle[30];
    int numero;
    int cod_postal ;
    char poblacion[30];
};
```

```
struct telefonos
{
    char casa[12] ={'\0'};
    char oficina[12]; ={'\0'};
    char movil[12] ={'\0'};
};
```

```
struct cliente
{
    char nombre[30];
    direccion casa;
    direccion oficina ;
    telefonos tlf;
    int edad;
} ;
```

cliente c;

```
...
c.nombre = "Ana Gonzalez";
c.casa.calle = "Av. Europa";
c.casa.numero = 45;
c.oficina.calle = "Pirineos";
c.tlf.oficina ="602.23.23.23";
c.tlf.casa = "91.123.45.67";
```

Ejercicio 01

Construir una aplicación en C++, una estructura que registre información de un producto, compuesto por los siguientes campos

- Código de producto (int)
- Descripción del producto (char 30)
- Tipo de Producto(char)
 - Tipo A: Electrohogar
 - Tipo B: Tecnologia
 - Tipo C: Deportes
- Precio del Producto (float)
- Stock(int)
- Valorización Total(float) Se calcula: precio del producto por el stock

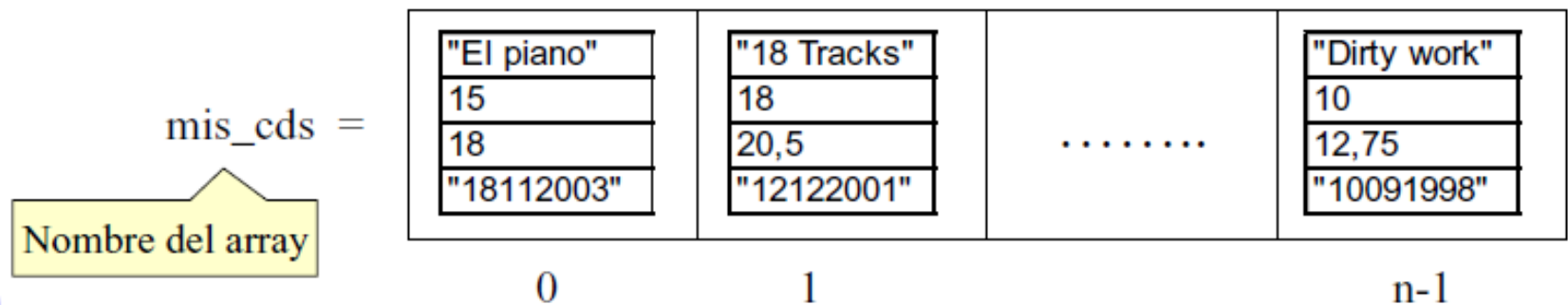
Mostrar por Consola los campos del producto, describiendo el tipo de producto

Arreglos con Estructura de Registros

TIPOS DE DATOS DEFINIDOS POR EL PROGRAMADOR: Estructuras

Arrays de estructuras:

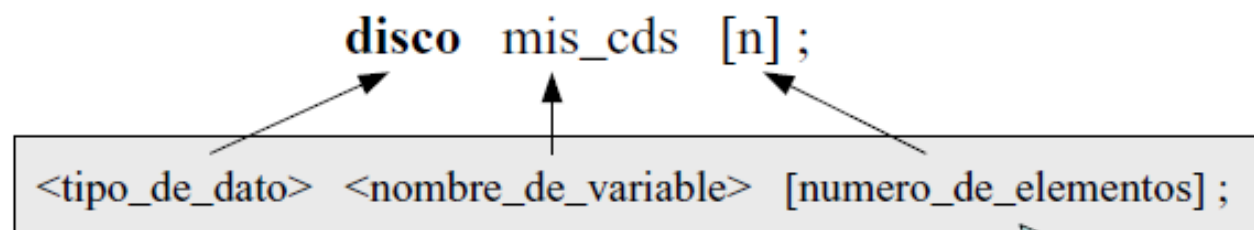
Supongamos que queremos guardar información de todos los discos que tenemos en nuestra casa. Con una variable de tipo disco, solo podemos guardar los datos de uno. Necesitaremos un array de discos:



- ➡ Tenemos una colección de elementos del mismo tipo.
- ➡ Se utilizan mucho los arrays de estructuras como método para almacenar datos en un archivo y leer datos de un archivo.

Arrays de estructuras. Declaración:

La declaración de un array de estructuras es igual a cualquier otro array:



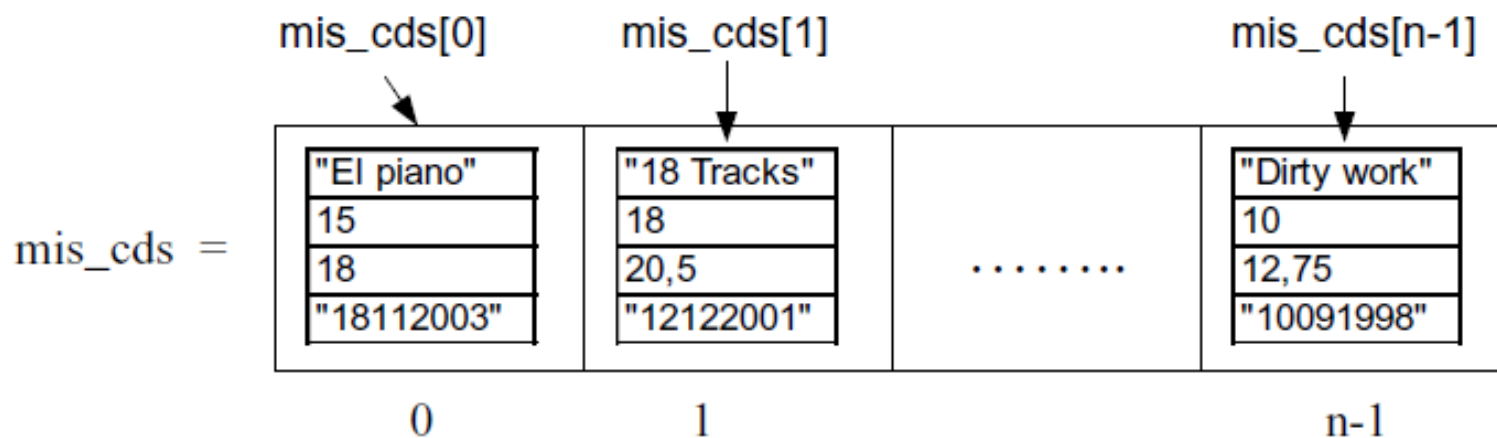
Sintaxis de la declaración
de un array

mis_cds =

<table><tr><td>"El piano"</td></tr><tr><td>15</td></tr><tr><td>18</td></tr><tr><td>"18112003"</td></tr></table>	"El piano"	15	18	"18112003"	<table><tr><td>"18 Tracks"</td></tr><tr><td>18</td></tr><tr><td>20,5</td></tr><tr><td>"12122001"</td></tr></table>	"18 Tracks"	18	20,5	"12122001"	<table><tr><td>"Dirty work"</td></tr><tr><td>10</td></tr><tr><td>12,75</td></tr><tr><td>"10091998"</td></tr></table>	"Dirty work"	10	12,75	"10091998"
"El piano"															
15															
18															
"18112003"															
"18 Tracks"															
18															
20,5															
"12122001"															
"Dirty work"															
10															
12,75															
"10091998"															
0	1		n-1												

Arrays de estructuras. Acceso a los elementos:

El acceso a los elementos se hace como en un array normal, se pone el nombre del array seguido del índice del elemento al que queremos acceder.



Ahora, si queremos acceder a los miembros de los elementos del array:

```
mis_cds[0].titulo = "El piano" ;  
mis_cds[1].precio = 20,5 ;
```

```
cin>> mis_cds[0].num_canciones ;  
cout << mis_cds[1].titulo ;
```

Ejercicio 02

Construir una aplicación en C++, una estructura que registre información de n libros, compuesto por los siguientes campos

- Código de libro(int)
- Título del libro(char 30)
- Año de publicación(int)
- Nombre de autor 1 (char 30)
- País del autor 1(int)
- Año nacimiento autor 1
- Regalia autor 1(float)... No llenar por teclado
- Nombre de autor 2 (char 30)
- País del autor 2 (char 10)
- Regalia autor 2(float))... No llenar por teclado
- Año nacimiento autor 2(int)
- Número de páginas (int)
- Email Contacto autor 1 (char 30)
- Email Contacto autor 2 (char 30)
- Email Editorial (char 30)
- Email Representante (char 30)
- Costo Libro (float)

Reestructurar el registro anterior, utilizando estructuras anidadas y resolver lo siguiente:

Las datos serán ingresados por teclado, cada autor debe recibir un 5% del costo del precio del libro.

Mostrar adecuadamente por consola los campos de cada libro.

Ejercicio

Elaborar un programa que permita obtener la informacion según la tabla adjunta

Codigo	Empleado	Tipo	Sueldo Basico	8%AFP	5%Seguro	%Bonif	Imp.Bonif	Sueldo Neto
1	CAMPOS	A	S/ 1,500.00	S/ 120.00	S/ 75.00			
2	CAPARÓ	B	S/ 2,500.00	S/ 200.00	S/ 125.00			
3	CARRASCO	A	S/ 3,000.00	S/ 240.00	S/ 150.00			
4	CASTLE	B	S/ 5,000.00	S/ 400.00	S/ 250.00			
5	CEDRON	A	S/ 3,500.00	S/ 280.00	S/ 175.00			
6	RIVERA	B	S/ 1,189.00	S/ 95.12	S/ 59.45			
7	SANCHEZ	A	S/ 2,866.00	S/ 229.28	S/ 143.30			
8	CHAVEZ	B	S/ 3,500.00	S/ 280.00	S/ 175.00			

Considerar lo siguiente:

- Permitir ingresar un numero N de empleados
- Tipo de Empleados A: Nombrados y B:Contratados
- AFP es el descuento del 8% de la remuneración básica
- Seguro es el 5% de dcto de la remuneración básica
- % Bonificacion. 12% Para Nombrados y 8% para contratados
- Sueldo Neto = Basico – AFP – Seguro + Bonificacion

Tipo	Cantidad	Importe Sueldo Neto
A		
B		