

# Programación 1

Tema 7. Tipos de datos estructurados: registros

**Grado en Ingeniería Informática**

# Objetivos / Competencias

2

1. Comprender el concepto del tipo de dato registro
2. Aprender a definir y utilizar estructuras de datos complejas, anidando tipos de datos estructurados
3. Aprender a manejar tipos de datos registro en lenguaje C

# Índice

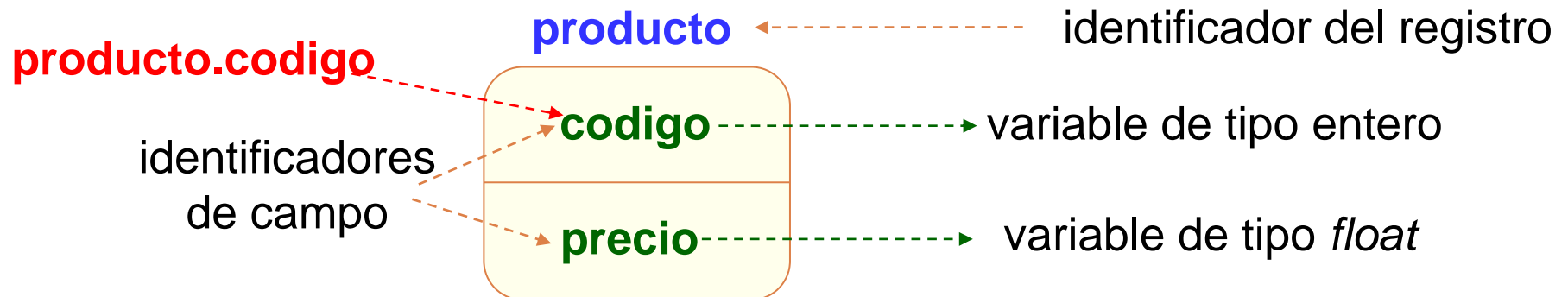
3

1. El tipo registro
2. Arrays de registros
3. Ejemplos
4. Fuentes de información

# El tipo registro

4

- El **tipo registro** es una estructura de datos en la que se almacena una colección finita de elementos, no necesariamente homogéneos (no tienen por qué ser todos del mismo tipo)
  - En un registro se agrupan atributos de una entidad
- Cada uno de los elementos de un registro se denomina **campo**
  - Para referirse a un determinado elemento de un registro se deberá utilizar el identificador del registro, seguido de un punto '.' y del identificador del campo correspondiente
- Ejemplo de registro con dos elementos :



# Ejemplos de registros

5

## Dirección

<b>calle</b>	array de char
<b>código postal</b>	array de char
<b>ciudad</b>	array de char

## Fecha

<b>día</b>	entero
<b>mes</b>	entero
<b>año</b>	entero

## Libro

<b>autor</b>	array de char
<b>título</b>	array de char
<b>prestado</b>	boolean

## Empleado

<b>nombre</b>	array de char
<b>Nº seguridad social</b>	array de char
<b>sueldo</b>	float
<b>Dirección</b>	Registro
<b>Fecha nacimiento</b>	Registro

# Definición de un registro en lenguaje C

6

- ▣ Para poder utilizar una variable de tipo registro primero tenemos que definir dicho tipo.

```
struct nombre{  
    tipo_campo1 nombre_campo1;  
    tipo_campo2 nombre_campo2;  
    ...  
};
```

- **nombre**: nombre del tipo registro definido; puede ser cualquier identificador válido
- **tipo\_campo**: tipo de cada uno de los campos del registro
- **nombre\_campo**: nombre de cada campo del registro; ¡Puede haber tantos campos como se necesario!

# Ejemplos de declaración de registros

7

```
struct TProducto {  
    int  codigo;  
    float precio;  
};
```

```
TProducto p1, p2;
```

p1 y p2 son variables del tipo  
registro TProducto

```
struct TControl{  
    int cod;  
    bool testigo[10];  
};
```

```
TControl c1;
```

Un array de booleanos  
dentro de un registro

```
struct TNif{  
    int numero;  
    char letra;  
};
```

```
struct TSocio{  
    TNif nif;  
    char nombre [30];  
};
```

```
TSocio socio1;
```

Registros  
anidados

# Inicialización y acceso a un registro

8

- ▣ Para inicializar un registro tenemos que inicializar cada uno de sus campos, accediendo a cada uno de ellos.
- ▣ Para acceder a un campo del registro utilizamos el **operador** “.”

```
nombre_registro.nombre_campo ;
```

- `nombre_registro`: identificador que denota el registro
- `nombre_campo` : indica el nombre del campo del registro

Código de un producto *p1*: `p1.codigo`;

Primer testigo de un control *c1*: `c1.testigo[0]`;

Letra del NIF de un socio *socio1*: `socio1.nif.letra`;



# Ejemplos de inicialización y acceso a un *registro*

9

```
struct TProducto{  
    int  codigo;  
    float precio;  
};
```

```
TProducto p1, p2;
```

```
p1.codigo = 3;  
p1.precio = 34.8;  
p2 = p1;
```

🔥 La asignación de registros está permitida en C

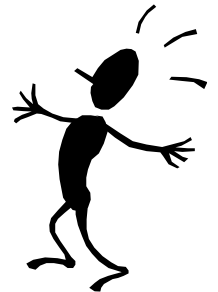


```
struct TLibro{  
    bool prestado;  
    char autor [30];  
    char titulo [50];  
    TFecha fec_pre;  
};
```

```
TLibro libro1, libro2;
```

🔥 Se pueden anidar registros, es decir, el campo de un registro puede ser otro registro

```
strcpy(libro1.autor , "Quevedo");  
cin.getline(libro1.titulo, 50-1);  
libro1.prestado = true;  
libro1.fec_pre.dia = 16;  
libro1.fec_pre.mes = 11;  
libro1.fec_pre.anyo = 2010;
```



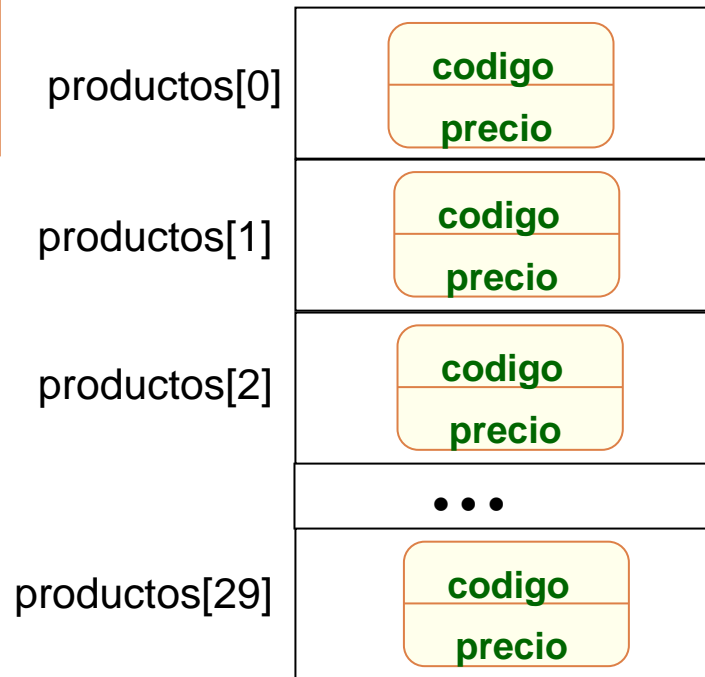
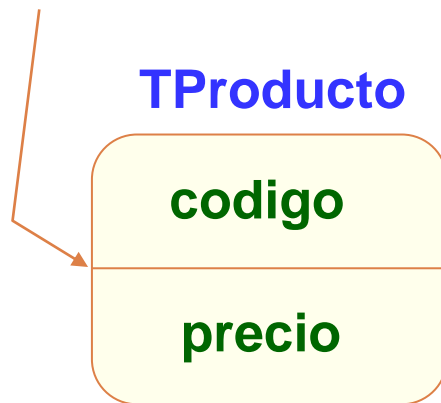
# Arrays de registros

10

- ▣ Cada elemento del array es un registro

```
struct TProducto{  
    int  codigo;  
    float precio;  
};
```

```
typedef TProducto TListaProductos [30];  
TListaProductos productos;
```



🔥 ¿Cómo accederías al código del tercer producto?

**productos[2].codigo**

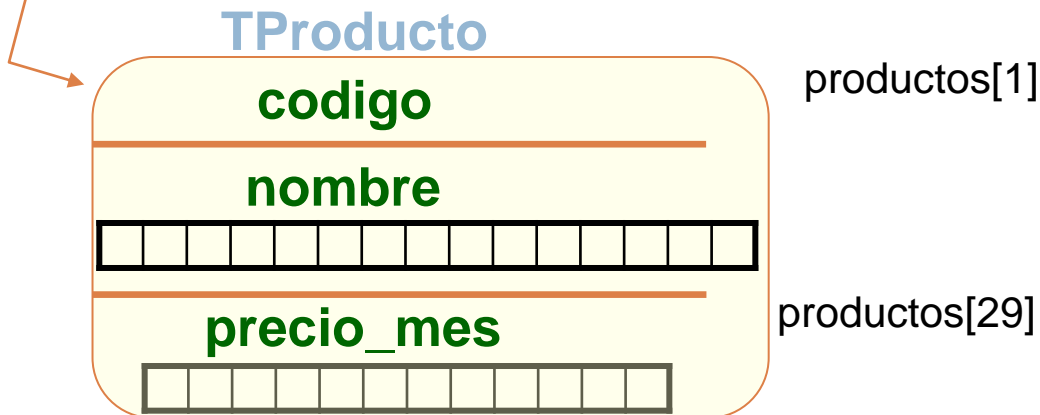
# Arrays de registros

11

- El campo de un registro puede ser a su vez un array

- Ejemplo

```
struct TProducto {  
    int  codigo;  
    char nombre [15]  
    float precio_mes[12];  
};
```

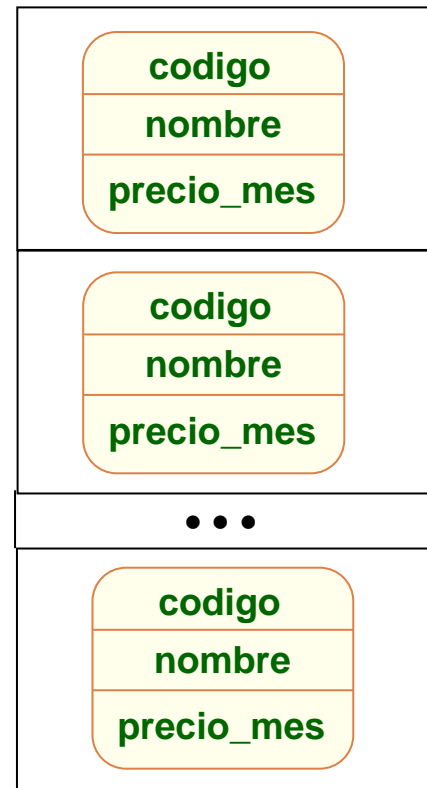


TProducto productos [30];

productos[0]

productos[1]

productos[29]



¿Cómo accederías al precio en el mes de agosto del quinto producto?

# Ejemplo 1

12

Definir las estructuras de datos necesarias para procesar la siguiente información:

- Una empresa de alquiler de vehículos desea gestionar la información acerca de los vehículos que tiene (no más de 200). Concretamente: matrícula, marca, modelo, fecha de compra y km mensuales realizados para todo el año, con la finalidad de obtener los vehículos que realizan más kilómetros de media al año (podrá ser uno solo o muchos con la misma media)

# Posibles estructuras de datos...

13

Definir las estructuras de datos necesarias para procesar la siguiente información:

- Una empresa de alquiler de vehículos desea gestionar la información acerca de los **vehículos** que tiene (no más de 200). Concretamente: matrícula, marca, modelo, fecha de compra y km mensuales realizados para todo el año, con la finalidad de obtener los vehículos que realizan más kilómetros de media al año (podrá ser uno solo o muchos con la misma media)

vehículo

matrícula → cadena de caracteres

marca → cadena de caracteres

modelo → cadena de caracteres

fecha de compra → Registro: día, mes, año

km mensuales x 12 meses → array con 12 enteros

**array con 200  
vehículos**

# Nuestro objetivo es...

14

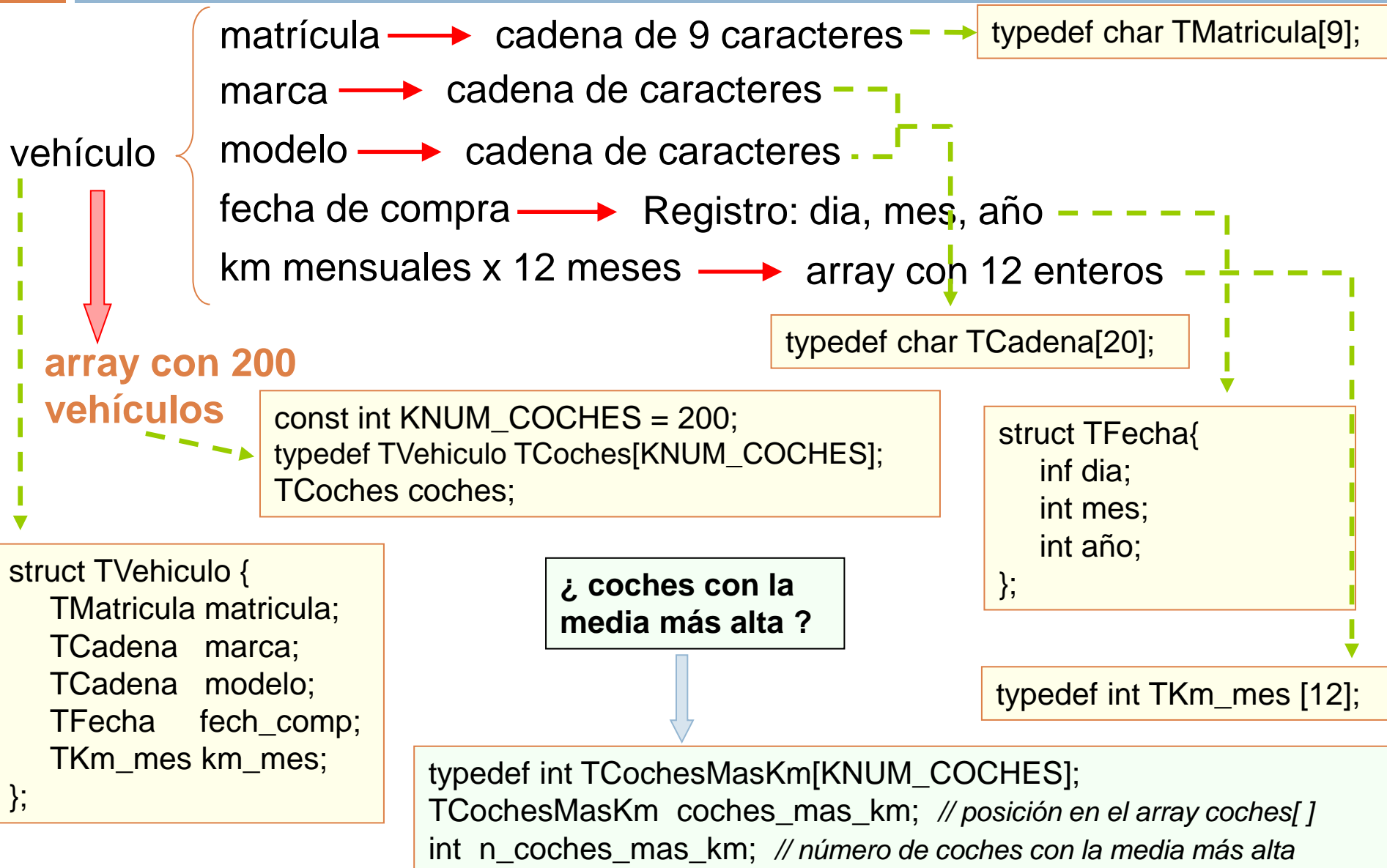
Definir las estructuras de datos necesarias para procesar la siguiente información:

- Una empresa de alquiler de vehículos desea gestionar la información acerca de los vehículos que tiene (no más de 200). Concretamente: matrícula, marca, modelo, fecha de compra y km mensuales realizados para todo el año, con la finalidad de obtener los vehículos que realizan más kilómetros de media al año (podrá ser uno solo o muchos con la misma media)

array con los índices de los vehículos con media más alta ( $\leq 200$ )

# Diseñamos los datos...

15



# Ejemplo 2

16

Definir las estructuras de datos necesarias para procesar la siguiente información:

- En una planta de fabricación de lavadoras quieren establecer un control de calidad informatizado de sus prototipos. Cada electrodoméstico viene caracterizado por un código numérico y una serie de características: capacidad (en kilos), modelo, tipo de carga (superior/frontal) y el resultado de los 10 controles a los que ha sido sometido. El control sólo tiene dos posibilidades: se ha pasado o no. Además, hay que saber qué revisor ha efectuado cada control. Un revisor puede realizar varios controles sobre el mismo aparato. De cada revisor se tiene la siguiente información: código numérico, nombre y departamento al que pertenece. La planta fabrica 25 prototipos al año

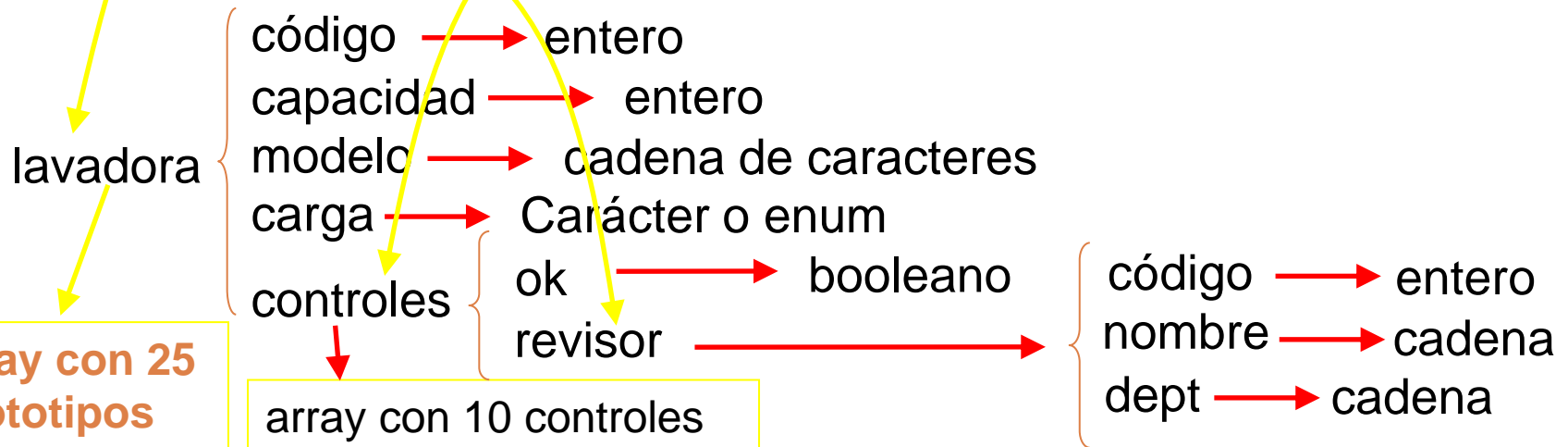


# Revisamos bien el texto ...

17

Definir las estructuras de datos necesarias para procesar la siguiente información:

- En una planta de fabricación de lavadoras quieren establecer un control de calidad informatizado de sus prototipos. Cada electrodoméstico viene caracterizado por un código numérico y una serie de características: capacidad (en kilos), modelo, tipo de carga (superior/frontal) y el resultado de los 10 controles a los que ha sido sometido. El control sólo tiene dos posibilidades: se ha pasado o no. Además, hay que saber qué revisor ha efectuado cada control. Un revisor puede realizar varios controles sobre el mismo aparato. De cada revisor se tiene la siguiente información: código numérico, nombre y departamento al que pertenece. La planta fabrica 25 prototipos al año



# Diseñamos los datos...

18

