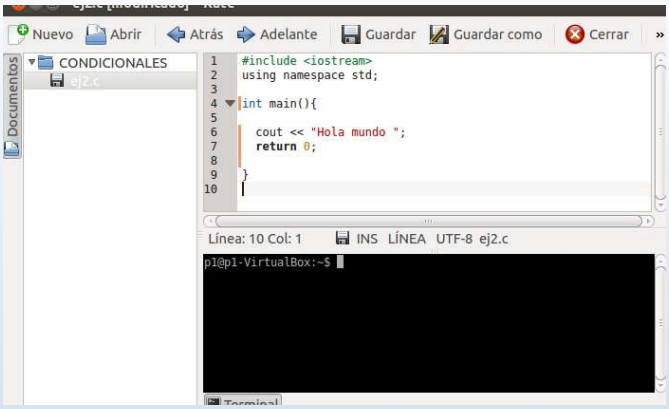


Programación 1

Tema 7. Tipos de datos estructurados: registros

Grado en Ingeniería Informática

Objetivos / Competencias

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

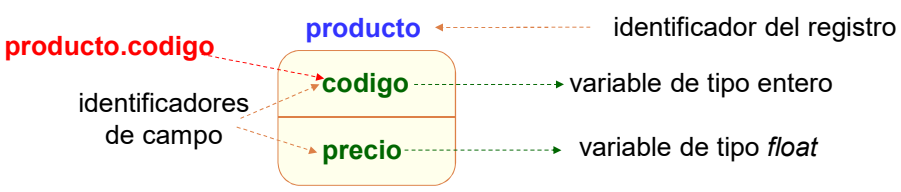
3

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	
calle	array de char
código postal	array de char
ciudad	array de char

Libro	
autor	array de char
titulo	array de char
prestado	boolean

Fecha	
dia	entero
mes	entero
año	entero

Empleado	
nombre	array de char
Nº seguridad social	array de char
sueldo	float
Dirección	Registro
Fecha nacimiento	Registro

Definición de un registro en lenguaje C

6

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

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

  - 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

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

TProducto  p1, p2;
```

p1 y p2 son variables del tipo registro *TProducto*

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

TControl c1;
```

Un array de booleanos dentro de un registro

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

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

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

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

TProducto  p1, p2;
```

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

- La asignación de registros está permitida en C

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

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

TLibro libro1, libro2;
```

```
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;
```

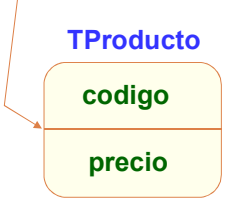
Arrays de registros

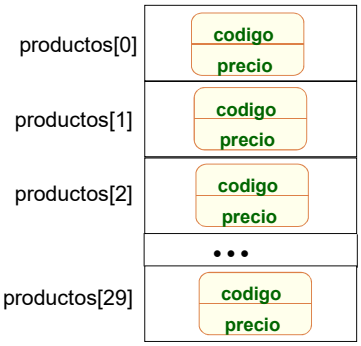
10

- Cada elemento del array es un registro

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

```
typedef struct {
    int    codigo;
    float precio;
}Tproducto ;
```





- ¿Cómo accederías al código del tercer producto?

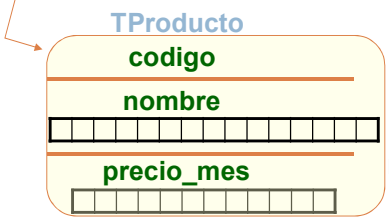
productos[2].codigo

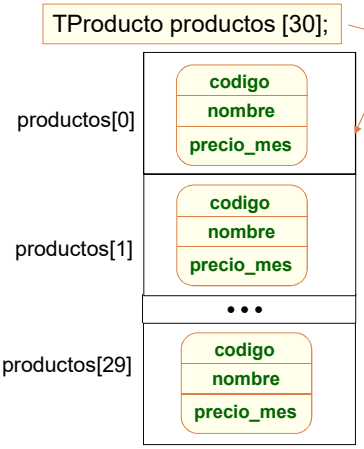
Se pueden anidar tipos de datos estructurados (I)

11

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

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





¿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: dia, 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 (≤ 200)

Diseñamos los datos...

15

vehículo

- matrícula → cadena de 9 caracteres
- marca → cadena de caracteres
- modelo → cadena de caracteres
- fecha de compra → Registro: dia, mes, año
- km mensuales x 12 meses → array con 12 enteros

```
typedef char TMatricula[9];
typedef char TCadena[20];
typedef int TKm_mes [12];
```

array con 200 vehículos

```
const int KNUM_COCHES = 200;
typedef TVehiculo TCoches[KNUM_COCHES];
TCoches coches;
```

```
typedef struct {
    inf dia;
    int mes;
    int año;
}TFecha;
```

```
typedef struct {
    TMatricula matricula;
    TCadena   marca;
    TCadena   modelo;
    TFecha    fech_comp;
    TKm_mes km_mes;
}TVehiculo;
```

¿ coches con la media más alta ?

```
typedef int TCochesMasKm[KNUM_COCHES];
TCochesMasKm  coches_mas_km;  // posición en el array coches[ ]
int  n_coches_mas_km;  // número de coches con la media más alta
```

Ejemplo 2

16

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

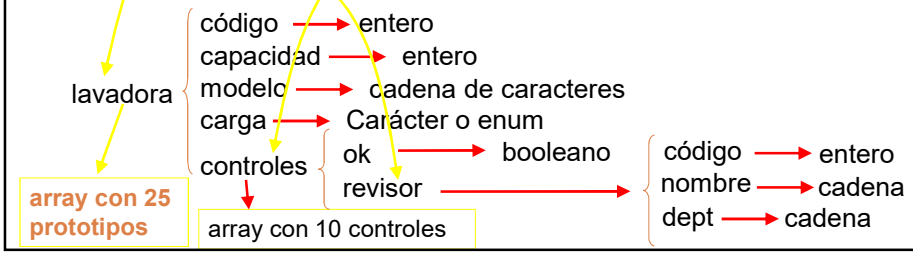
- En una planta de fabricación de lavadoras quieren establecer un control de calidad informatizado de sus prototipos. Cada electrodoméstico viene caracterizado por un código numérico y una serie de características: capacidad (en kilos), modelo, tipo de carga (superior/frontal) y el resultado de los 10 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

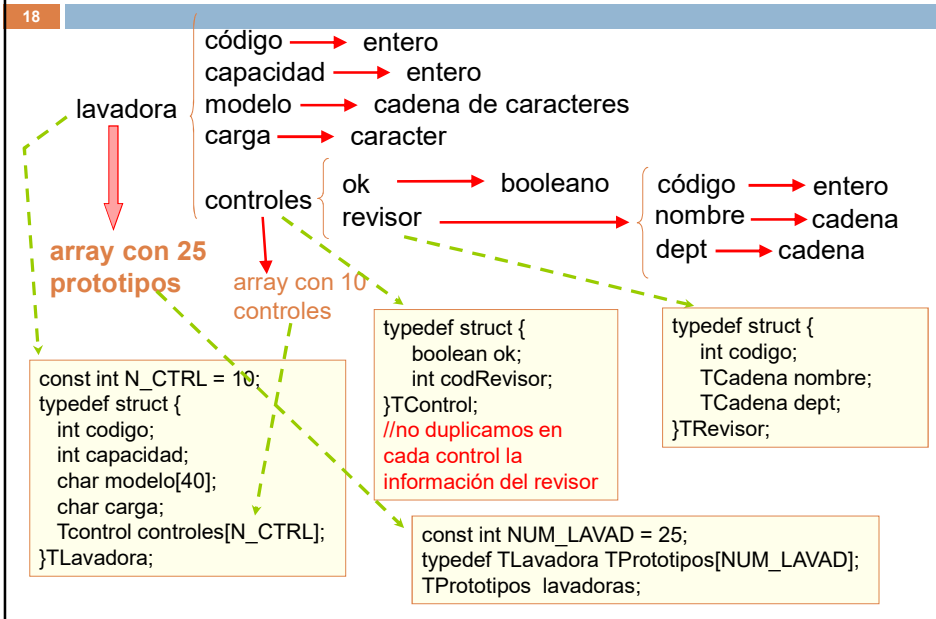
Definir las estructuras de datos necesarias para procesar la siguiente información:

- 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



Bibliografía Recomendada

Fundamentos de Programación
Jesús Carretero, Félix García, y otros
Thomson-Paraninfo 2007. ISBN: 978-84-9732-550-9

- Capítulo 9 (Apartados 9.1.1; 9.1.2; 9.3)

Problemas Resueltos de Programación en Lenguaje C
Félix García, Alejandro Calderón, y otros
Thomson (2002) ISBN: 84-9732-102-2

- Capítulo 2 (Apartados 7.1.1; 7.1.2; 7.2)

Resolución de Problemas con C++
Walter Savitch
Pearson Addison Wesley 2007. ISBN: 978-970-26-0806-6

- Capítulo 6 (Apartado 6.1)