```
♦ Atrás
♦ Adelante
Guardar
✓ Guardar como
                                                                          (S) Cerrar
 Nuevo Abrir
                               #include <iostream>
    CONDICIONALES
Documentos
                               using namespace std;
      ▼ int main(){
                                 cout << "Hola mundo ";
                                 return 0;
                           8
                           9
                          10
                           Línea: 10 Col: 1
                                           INS LÍNEA UTF-8 ej2.c
                          p1@p1-VirtualBox:~$
                          Torminal
```

Programación 1

Tema 7. Tipos de datos estructurados: registros

Grado en Ingeniería Informática

Objetivos / Competencias

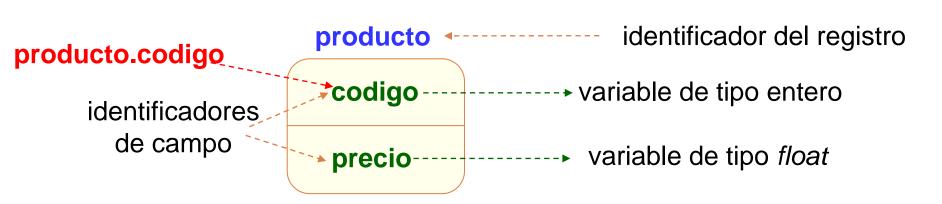
- Comprender el concepto del tipo de dato registro
- Aprender a definir y utilizar estructuras de datos complejas, anidando tipos de datos estructurados
- Aprender a manejar tipos de datos registro en lenguaje C

Índice

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

El tipo registro

- El tipo registro es una estructura de datos en la que se almacena una colección <u>finita</u> de elementos, <u>no</u> necesariamente <u>homogéneos</u> (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

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

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

```
struct TProducto {
    int codigo;
    float precio;
};
TProducto p1, p2;
```

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

TControl c1;
```

Registros anidados

```
p1 y p2 son variables del tipo registro TProducto
```

Un array de booleanos dentro de un registro

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

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

TSocio socio1;
```

Inicialización y acceso a un registro

- Para inicializar un registro tenemos que inicializar cada uno de sus campos, accediendo a cada uno de ellos.
- Para <u>acceder</u> 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 socio 1: socio 1.nif.letra;

Ejemplos de inicialización y acceso a un registro

```
struct TProducto{
                                  p1.codigo = 3;
   int codigo;
                                  p1.precio = 34.8;
   float precio;
                                  p2 = p1;
};

    La asignación de registros

TProducto p1, p2;
                                              está permitida en C
                                  Se pueden anidar registros,
                                  es decir, el campo de un
                                  registro puede ser otro registro
struct TLibro{
    bool prestado;
    char autor [30];
                                   strcpy(libro1.autor ,"Quevedo");
    char titulo [50];
                                   cin.getline(libro1.titulo, 50-1);
    TFecha fec_pre;
                                   libro1.prestado = true;
};
                                   libro1.fec_pre.dia = 16;
TLibro libro1, libro2;
                                   libro1.fec_pre.mes = 11;
                                   libro1.fec_pre.anyo = 2010;
```

Arrays de registros

Cada elemento del array es un registro

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

TProducto

codigo

precio

productos[29]

typedef TProducto TListaProductos [30]; TListaProductos productos;

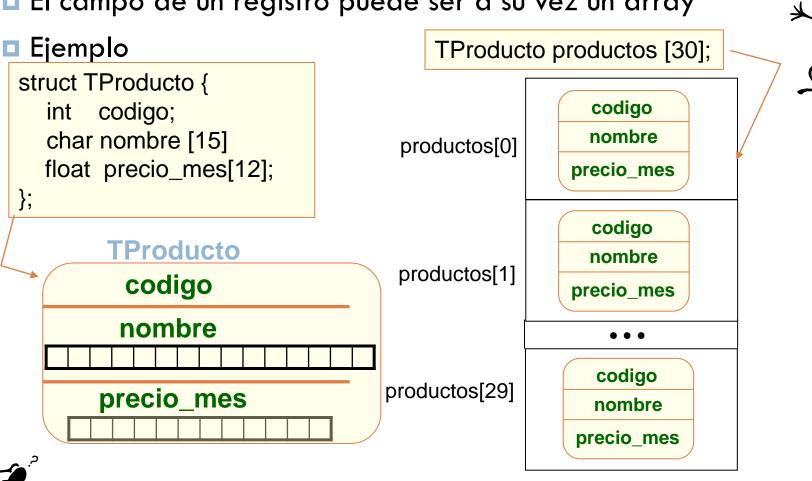
codigo productos[0] precio codigo productos[1] precio codigo productos[2] precio codigo precio



¿Cómo accederías al código del tercer producto? productos[2].codigo

Arrays de registros

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



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

Ejemplo 1

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...

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...

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
            matrícula → cadena de 9 caracteres - → typedef char TMatricula[9];
           marca --- cadena de caracteres
           modelo --- cadena de caracteres - -
vehículo
           km mensuales x 12 meses → array coh 12 enteros
                                           typedef char TCadena[20];
  array con 200
  vehiculos
              const int KNUM_COCHES = 200;
                                                       struct TFecha{
              typedef TVehiculo TCoches[KNUM_COCHES];
                                                          inf dia;
               TCoches coches;
                                                          int mes:
                                                          int año;
struct TVehiculo {
                               ¿ coches con la
                                                       };
  TMatricula matricula:
                               media más alta?
  TCadena marca:
  TCadena modelo;
                                                       typedef int TKm_mes [12];
  TFecha fech_comp;
  TKm mes km mes;
                       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

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 ...

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

código entero
capacidad entero
modelo cadena de caracteres
carga Carácter o enum
controles ok booleano
revisor cadena
array con 25
prototipos controles array con 10 controles

Diseñamos los datos...

```
18
                código --- entero
                capacidad --- entero
                modelo --- cadena de caracteres
    lavadora
                carga --- caracter
                             ok
                                            booleano
                                                          código → entero
                controles
                                                          nombre ---- cadena
                             revisor
                                                          dept → cadena
  array con 25
  prototipos
                   array con
                   controles
                                 struct TControl{
                                                            struct TRevisor{
                                    boolean ok;
                                                              int codigo;
                                                              TCadena nombre;
 const int N CTRL = 10;
                                    int codRevisor;
 struct TLavadora{
                                                              TCadena dept;
                                                           };
   int codigo;
                                 //no duplicamos en
   int capacidad;
                                 cada control la
   char modelo[40];
                                 información del revisor
   char carga;
   Tcontrol controles[N_CTRL];
                                    const int NUM_LAVAD = 25;
 };
                                    typedef TLavadora TPrototipos[NUM_LAVAD];
                                     TPrototipos lavadoras;
```