LECCIÓN 1:Tipos de datos estructurados (II)

María de la Paz Guerrero Lebrero
Curso 2014 / 2015
Grado en Matemáticas
maria.guerrero@uca.es

UCA
Universidad de Cádiz

Índice

- Extensión de los vectores: Matrices (nxm)
- Estructuras.
- Tipos de datos definidos por el usuario.
- Tipos de datos enumerados.
- Matrices multidimensionales.

Extensión de los vectores: matrices (nxm)

- Definición
- Representación
- Acceso a un elemento
- Almacenamiento interno (por filas)
- Recorrido de una matriz
- Operaciones con matrices
- Un ejemplo
- Ejercicios

Definición

- Es una E.D. similar al vector, es decir, constituida por varios datos del mismo tipo, agrupados bajo un nombre común, pero donde el acceso se realiza por 2 índices.
- El acceso a cada uno de los elementos se realiza indicando su posición relativa en la matriz (fila y columna).
- Lo declararemos de la siguiente forma:

```
<tipo> <nombre>[<numfilas>][<numcols>];
```

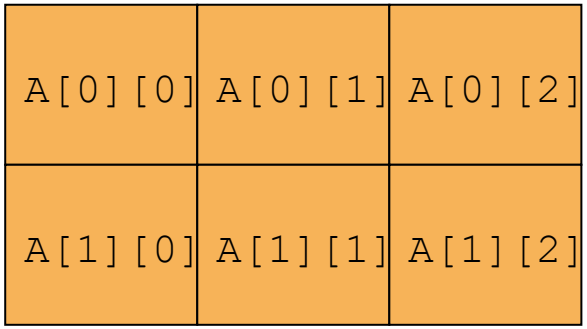
- Ejemplos:

```
int A[10][20];
float V[10][5];
```

Representación

- Sea la declaración

```
int A[2][3];
```



- La primera dimensión es la fila, y la segunda la columna
- Al igual que en los vectores, los rangos permitidos dependen del lenguaje. En C empezaremos en 0.

Acceso a un elemento

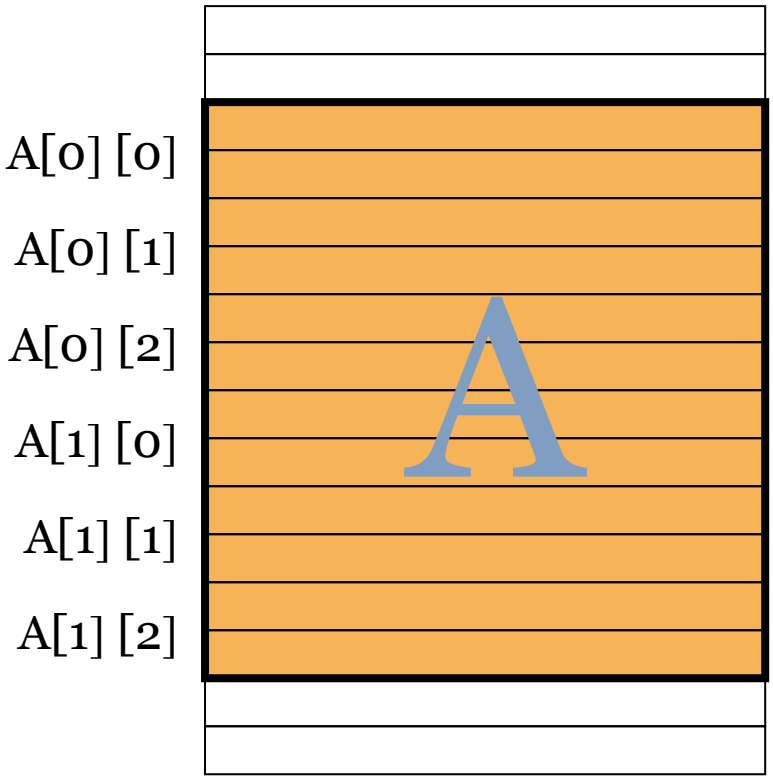
- El acceso a un elemento de una matriz se realiza por su posición, indicándose la fila y la columna donde se encuentra.

- Para acceder a un elemento se escribe el nombre de la variable, la fila entre corchetes, y la columna entre corchetes de la siguiente forma:

```
<nombre de variable>[<posición>][<posición>]
```

Almacenamiento interno (por filas)

```
int A[2][3];
```



Recorrido de una matriz

- Dado que el acceso a los elementos de una matriz se hace elemento a elemento, y que se conoce a priori su tamaño, es muy frecuente acceder a sus elementos con un doble bucle FOR.

- Ejemplo : Asignar 0 a todos los elementos de una matriz de tamaño MxN.

```
for(i=0;i<M;i++)
   for (j=0;j<N;j++)
      A[i][j]=0;
```

Operaciones con matrices

- Las operaciones con matrices permitidas dependen del lenguaje.
- Normalmente, no se permite ninguna operación, y de existir alguna permitida, ésta es la asignación de una matriz a otra del mismo tipo
- En C no se permite ninguna operación sobre matrices completas

Un ejemplo

- Sean las matrices A_{MxN} y B_{NxP}, calcular C_{MxP} como el producto de la matriz A por la matriz B:

```
for (i=0;i<M;i++)
{
    for (j=0;j<P;j++)
    {
        suma=0;
        for(k=0;k<N;k++)
            suma=suma+A[i][k]*B[k][j];
        C[i][j]=suma;
    }
}
```

Ejercicios

1. Inicializa una matriz de 3x3 a los 9 primeros números naturales.
2. Inicializa una matriz de 3x3 a los primeros nueve números primos.
3. Crea una nueva matriz que sea la suma de las dos anteriores.
4. Calcula el valor promedio de los elementos de la matriz del ejercicios 3.

Estructuras

- Definición
- El operador punto (.)
- El operador flecha (->)

Estructuras

- Es un tipo de dato definido por el programador.
- Está formado por un conjunto de N elementos (que pueden ser de distinto tipo) denominados campos, agrupados bajo el mismo nombre.
- Los campos de las estructuras pueden ser a su vez otras estructuras.

Estructuras

- Primera forma de declarar estructuras:
  - Se declara la estructura y la variable de tipo estructura al mismo tiempo.

```
struct <nombre de la estructura>
{
        <tipo> <nombre del campo>;
         <tipo> <nombre del campo>;
        ...
} <variable de tipo estructura>;
```

Estructuras

- Ejemplo:

```
#include <stdio.h>

void main (void)
{
        struct persona
        {
                char nombre [30];
                char apellidos [30];
                char direccion [30];
                int edad;
                char dni [30];
        } pepe;
}...
```

Pepe es una variable de tipo struct persona

Estructuras

- Segunda forma de declarar estructuras:
  - Se declara primero la estructura y luego la variable de tipo estructura.
  - **VENTAJA**: se pueden declarar otras variables del mismo tipo sin repetir la estructura.

```
struct <nombre de la estructura>
{
        <tipo> <nombre del campo>;
         <tipo> <nombre del campo>;
        ...
} ;
struct <nombre de la estructura> <variable de tipo estructura>;
```

Estructuras

- Ejemplo:

```
#include <stdio.h>

void main (void)
{
        struct persona
        {
                char nombre [30];
                char apellidos [30];
                char direccion [30];
                int edad;
                char dni [30];
        } ;
        struct persona pepe;
        struct persona amigos [10];
}
```

Operador punto (.)

- Acceso a los campos de la estructura:
  - A través de la variable de tipo estructura mediante el operador punto.

<variable de tipo estructura> . <nombre del campo>;

Operador punto (.)

- Ejemplo:

```
struct persona pepe;
strcpy(pepe.nombre, “Jose”);
strcpy(pepe.apellidos, “Garcia Perez”);
strcpy(pepe.direccion, “Calle Narvaez n. 11 5ª A”);
pepe.edad = 32;
strcpy(pepe.dni, “12345678R”);
```

Operador flecha (->)

- Acceso a los campos de la estructura:
  - A través de un puntero a la variable de tipo estructura mediante el operador flecha.

<puntero a la variable de tipo estructura> -> <nombre del campo>;

Operador flecha (->)

- Ejemplo:

```
struct persona *juan;
strcpy(juan->nombre, “Juan Antonio”);
strcpy(juan->apellidos, “Moreno Sanchez”);
strcpy(juan->direccion, “Calle Santo Dmingo n. 6 3ª D”);
juan->edad = 28;
strcpy(juan->dni, “98765432L”);
```

Inicialización de estructuras

- Las inicializaciones del ejemplo anterior también se puede realizar de la siguiente forma:

```
struct persona pepe = { "José", "García Pérez",
                        "Calle Narvaez  n.11 5ºA",  32,
                        "12345678R"};
```

Ejercicios

1. Crear un programa que contenga los siguientes elementos:
   a) Una estructura que represente puntos formada por sus coordenadas x e y.
   b) Una estructura que represente triángulos utilizando tres vértices (cada vértices es un elemento de tipo punto)
   c) Una función llamada **mostrarCoordenadas** que muestre por pantalla las coordenadas de los puntos en este formato: *(2,3) (la x es el dos y la y el tres)*
   d) Una función llamada **InsertarCoordenadas** que permita introducir las coordenadas del punto que se envía como parámetro.
   e) Una función llamada **distanciaPuntos** que devuelva la distancia entre dos puntos (que recibe como parámetros de la función)
   f) Una función llamada **perimetroTriangulo** que permita escribir el perímetro de un triángulo dado.

Tipos de datos definidos por el usuario

- Definición
- Ejemplos

Typedef

- La palabra reservada **typedef** se utiliza para asignar un alias (otro nombre) a un tipo.
- Crea ningún tipo nuevo, solo define un nuevo identificador para un tipo que ya tiene su propio identificador.

```
typedef <tipo><alias>
```

Typedef

- Ejemplo:

```
typedef  char[20]  nombre;
nombre n1, n2, n3;
typedef struct
{
        char nombre[30];
        char apellidos[30];
        char dirección [30];
        int edad;
        char dni[30];
} TPersona;
void main
{
        TPersona pepe, juan;
        TPersona amigos[20];
}
```

Tipos de datos enumerados

- Definición
- Ejemplo

enum

- Una enumeración es un conjunto de valores enteros constantes con nombre.

enum <identificador>{lista de valores enumerados }

enum

- Ejemplo

```
enum semana{lunes, martes, miercoles, jueves, viernes,
            sábado, domingo};

enum meses {enero, febrero, marzo, abril, mayo, junio,
            julio, agosto, septiembre, octubre,
            noviembre,  diciembre};

enum figuras {círculo, cuadrado, triángulo};
```

Ejercicios

1. Diseña el tipo de dato necesario para representar una lista de pacientes. Los datos necesarios son:
    1. Nombre
    2. Apellidos
    3. D.N.I
    4. Nº de la seguridad social
    5. Dirección
    6. Indicación si es alérgico a los ácaros, a las gramíneas o epitelio de gato.

Ejercicios

2. Implementa las funciones necesarias para:
   1. Insertar datos de un paciente.
   2. Modificar el registro de un paciente
   3. Mostrar la información de un paciente.
   4. Eliminar un paciente

Matrices multidimensionales

- Definición
- Ejemplo

Matrices multidimensionales

- Las matrices de matrices (matrices multidimensionales), son simplemente matrices cuyos elementos son otras matrices.

```
<tipo ><nombre> [dim_1][dim_2][dim_3] ...[dim_n];
```

Matrices multidimensionales

- Ejemplo

```
int dias[2][12] = {
  {31,28,31,30,31,30,31,31,30,31,30,31},
  {31,29,31,30,31,30,31,31,30,31,30,31} };

char lista[5][4] = { "Cat", "Bar", "Cab", "Lab", "Tab" };

int  numeros[2][5][3] = {
        { {1,2,3}, {4,5,6}, {7,8,9}, {2,3,4}, {6,7,8}},
        { {9,8,7}, {7,6,5}, {6,5,4}, {4,3,2}, {3,2,1}} };
```