Autoestudio 11: Matrices

Un **arreglo** lo definimos como **la relación entre un nombre y un conjunto de localidades**. Decimos que la estructura de datos que definimos en el arreglo es de una sola dimensión ya que utilizamos un solo valor para identificar a cada localidad (0,1,2,...,n-1).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |

Una **matriz** también es una colección de localidades asociadas a un nombre, sólo que los datos se organizan en **dos dimensiones**, es decir, **una matriz es un arreglo multidimensional** o un arreglo de arreglos. Por ello, para hacer referencia a una localidad de la matriz se necesitan de dos números:

\* El número de renglón

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 0 |  |  |  |  |  |  |  |  |  |  |
| 1 |  |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |

\* El número de columna

De la misma manera que en los arreglos, la numeración de renglones y de columnas inicia desde **0**.

Si por ejemplo definimos una matriz de **6** renglones y **6** columnas, el primer elemento de la colección se encontraría en:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 |
| 0 | **x** |  |  |  |  |  |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  |  | **X** |

* + **renglón 0**
  + **columna 0**

Y el último elemento se encontraría en:

* + **renglón 5**
  + **columna 5**

La definición de variables de tipo matriz es similar a los arreglos, la forma general de declarar una variable matriz es la siguiente:

**tipo nombre[#renglones][#columnas];**

en donde **tipo** es un tipo de datos (int, float, char), **nombre** es el nombre de la variable, **#renglones** es el número de renglones que tendrá la matriz y **#columnas** el número de columnas de la matriz.

Una vez que hemos declarado la variable matriz para tener acceso a los valores indicamos el **renglón** y la **columna** de dicha localidad. Para esto debemos recordar que los renglones están numerados de **0** a **r-1** (en donde r es el número de renglones del arreglo) y que las columnas están numeradas de **0** a **c-1** (en donde c es el número de columnas del arreglo).

Para hacer referencia a una localidad específica de la matriz debemos escribir el nombre de la variable y entre corchetes el número de renglón y el número de columna de la localidad.

Si imaginamos que se cuenta con una matriz:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 0 | **a** | b | c | d | e | f | g | h | i | J |
| 1 | q | w | e | r | t | y | u | i | o | P |
| 2 | z | x | c | v | b | n | M | H | G | l |
| 3 | q | w | e | d | c | f | g | b | H | N |
| 4 | v | e | v | s | w | v | r | B | T | N |
| 5 | 0 | 9 | 9 | 8 | 6 | \_ | = | \* | / | - |
| 6 | b | b | . | , | m | . | ; | . | L | **k** |

**char M[7][10];**

La cual asumimos que está llena con los siguientes caracteres:

La manera como referenciamos a la primera localidad de la matriz M que se definió anteriormente es **M[0][0]** y la última localidad es **M[6][9].**

La forma de asignar un valor a una localidad específica de la matriz es la siguiente...

**nombre[renglon][columna] = valor;**

en donde **nombre** es el nombre de la variable matriz, **renglon** es el número del renglón de la localidad y **columna** el número de columna de la localidad de la matriz y **valor** es cualquier valor del tipo con que fue definida la matriz. Así por ejemplo para las siguientes instrucciones modifican el contenido de la matriz:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 0 | **A** | b | c | D | e | f | g | h | i | **%** |
| 1 | Q | w | e | R | t | **%** | u | i | o | P |
| 2 | Z | x | c | V | b | n | M | H | G | l |
| 3 | Q | w | e | **%** | c | f | g | b | H | N |
| 4 | V | e | v | S | w | v | r | **%** | T | N |
| 5 | 0 | 9 | 9 | 8 | 6 | \_ | = | \* | / | - |
| 6 | B | b | . | , | m | . | ; | . | L | **%** |

M[3][3]= ‘%’;

M[1][5]= ‘%’;

M[6][9]= ‘%’;

M[4][7]= ‘%’;

M[0][9]= ‘%’;

Así como usamos ciclos for para recorrer arreglos, también los usamos para recorrer matrices, sin embargo como una matriz es en realidad un arreglo anidado entonces usamos cilclos anidos. Esto significa que la estructura compañera de las matrices es **dos ciclos for anidados**,ya que un ciclo se utiliza para recorrer los renglones y otro para recorrer las columnas. La mayoría de los problemas de matrices tendrán un código similar al siguiente:

**int i, j, renglones = 7, columnas = 10;**

**for (i=0; i<renglones; i++){**

**for ( j=0;j<columnas; j++) {**

**printf(“ %c ”, M[i][j]);**

**}**

**printf(“ \n”);**

**}**

Este código imprimiría (observa el salto de línea **“\n”** que se inserta cada vez que se termina de recorrer el ciclo de adentro):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | b | c | D | e | f | g | h | i | **%** |
| Q | w | e | R | t | **%** | u | i | o | P |
| Z | x | c | V | b | n | M | H | G | l |
| Q | w | e | **%** | c | f | g | b | H | N |
| V | e | v | S | w | v | r | **%** | T | N |
| 0 | 9 | 9 | 8 | 6 | \_ | = | \* | / | - |
| B | b | . | , | m | . | ; | . | L | **%** |

Esta estructura de doble ciclo nos permite recorrer todas las localidades de la matriz, con el primer ciclo **for** se recorren los **renglones**, y con el segundo ciclo se recorren las **columnas**.

En el primer ciclo se recorre cada renglón, para ello la variable del ciclo (i) toma los valores:

**0**, **1**, **2**, ... , **renglones – 1** que son precisamente los números de cada renglón de la matriz.

En el segundo ciclo para el renglón **i**, se recorre cada **columna** de la matriz, para ello la variable del ciclo **( j )** toma los valores:

**0**, **1**, **2**, ... , **columnas-1**

que son precisamente los números de cada columna de la matriz.

## Definición de constantes

Cuando trabajamos con arreglos y matrices, es común que necesitemos repetir un valor múltiples veces para definir el tamaño de los arreglos y matrices. Cuando esto ocurre, es mejor definir una constante. En el lenguaje C, lo podemos hacer de la siguiente manera al inicio del programa justo después de los *includes*:

#define **NOMBRE\_DE\_LA\_CONSTANTE** *valor\_constante*

Por ejemplo:

#define **REN** 6

#define **COL** 9

Y después podemos utilizar los valores de las constantes definidas de la misma forma que se emplea el valor de una variable:

void imprimeMatriz(mat[REN][COL]) {

int i, j;

for (i = 0; i < REN; i++) {

for (j = 0; j < COL; j++) {

printf(“ %c ”, M[i][j]);

}

printf(“ \n”);

}

}

# Si deseas saber más puedes consultar, donde se explican los temas a mayor detalle:

(Joyanes Aguilar, L. y Zahonero Martínez, I. *Programación en C, C++, Java y UML*. México : McGraw Hill, 2010.) las siguientes secciones:

* Arreglos multidimensionales. (pp. 242)