UNIVERSIDAD DE MATANZAS

MATRICES O ARREGLOS BIDIMENSIONALES

Octubre / 2019

INTRODUCCIÓN A LA PROGRAMACIÓN
INGENIERÍA EN INFORMÁTICA

Objetivos

Caracterizar los arreglos bidimensionales o matrices como parte de la construcción de un programa computacional.

Objetivos

Declarar matrices , inicializarlas y hacer referencia a elementos individuales de las matrices.

Objetivos

Utilizar la instrucción **for** para iterar a través de las matrices.

Pasar matrices a los métodos como argumentos.

Sumario

- Arreglos
  - Declaración y creación
  - Uso de un inicializador
  - Instrucción **for** para iterar
  - Paso de matrices a los métodos

Bibliografía

- *Como programar en Java.*
- *Aprenda Java como si estuviera en primero.*

Matrices

En matemática, una matriz es un arreglo bidimensional de números. Dado que puede definirse tanto la suma como el producto de matrices, en mayor generalidad se dice que son elementos de un anillo. La notación de una matriz $\mathbf{A}$ tiene la forma:

Matrices

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

Matrices

Una matriz es un arreglo bidimensional de elementos (todos del mismo tipo de dato) ordenados en filas (o renglones) y columnas, donde una fila es cada una de las líneas horizontales de la matriz y una columna es cada una de las líneas verticales.

Matrices

A una matriz con n filas y m columnas se le denomina matriz n-por-m (escrito $n \times m$)

Matrices

Los tipos en Java se dividen en dos categorías: tipos primitivos y tipos de referencia. Las matrices son objetos, por lo que se consideran como tipos de referencia.

Matrices

Los elementos de una matriz pueden ser tipos primitivos o de referencia.
Para hacer referencia a un elemento específico en una matriz, debemos especificar el nombre de la referencia de la matriz y el número de la fila y el número de la columna del elemento en la matriz.

Matrices

La posición del elemento se compone formalmente de la **fila** y la **columna**.

Matrices

Un programa puede hacer referencia a cualquiera de estos elementos mediante **una expresión de acceso a una matriz** que incluye el nombre de la matriz, seguido por el índice-fila del elemento específico encerrado entre **corchetes** seguido igualemente por el índice-columna del elemento específico encerrado entre **corchetes** ([ ][ ]).

Matrices

Declaración y creación

Los objetos matrices ocupan espacio en memoria. Al igual que los demás objetos, los arreglos se crean con la palabra clave **new** .

Matrices

Declaración y creación

Para crear un objeto matriz, el programador especifica el tipo de cada elemento y el número de filas y columnas que se requieren para la matriz, como parte de una expresión para crear un arreglo que utiliza la palabra clave **new**.

Matrices

Declaración y creación

Sintaxis

```
<TD> [][] <nombre> = new <TD> [<fil>][<col>];
```

<TD>: Tipo de dato
<nombre>: Nombre de la matriz
<fil>: Cantidad de filas de la matriz
<col>: Cantidad de columnas de la matriz

Matrices

Declaración y creación

La siguiente declaración y expresión crea una matriz, que contiene 12 filas y 10 columnas **int**, y almacena la referencia de la matriz en la variable c

Código

```
int [][] c = new int[12][10];
```

Matrices

Declaración y creación

También puede realizarse en dos pasos, como se muestra a continuación:

```
int [][] c; // declara la variable matriz
c = new int[12][10]; /* crea la matriz; lo asigna a
    la variable tipo matriz*/
```

Matrices

Uso de un inicializador

Al igual que los arreglos unidimensionales, los arreglos multidimensionales pueden inicializarse mediante inicializadores de arreglos en las declaraciones.

Matrices

Uso de un inicializador

Un arreglo bidimensional b con dos fi las y dos columnas podría declararse e inicializarse con **inicializadores de arreglos anidados**, como se muestra a continuación:

```
int b[ ] [ ] = { { 1, 2 }, {3, 4} };
```

Matrices

Uso de un inicializador

Los valores del inicializador se agrupan por fila entre llaves. Así, 1 y 2 inicializan a b[0][0] y b[0][1] , respectivamente; 3 y 4 inicializan a b[1][0] y b[1][1] , respectivamente.

Matrices

Uso de un inicializador

Los arreglos multidimensionales se mantienen como arreglos de arreglos unidimensionales.

Matrices

Uso de un inicializador

Por lo tanto, el arreglo b en la declaración anterior está realmente compuesto de dos arreglos unidimensionales separados: uno que contiene los valores en la primera lista inicializadora anidada $1, 2$ y uno que contiene los valores en la segunda lista inicializadora anidada $3, 4$.

Matrices

Instrucción **for** para iterar

Utilizar las instrucciones **for** controladas por un contador para iterar a través de los elementos en un arreglo es muy común.
En caso de las matrices para iterar sobre ellas debes llevar dos contadores

Arreglos

Instrucción **for** para iterar

En este caso un contador para fila y otro para la columna.
Si para iterar en un arreglo hace falta un **for** en las matrices se utiliza un dos **for** anidados.

Matrices

Instrucción **for** para iterar

```
for(int i=0;i<filas;i++){
    for(int j=0;j<columnas;j++){
      System.out.println(a[i][j]);
    }
}
```

O puede ser de la siguiente manera:

```
for(int i=0;i<columnas;i++){
  for(int j=0;j<filas;j++){
    System.out.println(a[j][i]);
  }
}
```

Matrices

Instrucción **for** para iterar

En ambos casos se logra iterar sobre cada uno de los elementos de matriz. Pero con recorridos diferentes

Matrices

Paso de matrices a los métodos

Para pasar un argumento tipo matrices a un método, se especifica el nombre de la matriz sin corchetes. Por ejemplo, si matriz temperaturasPorHoraYDia se declara como

```
double [][] temperaturasPorHoraDia = new double
    [24][365];
```

entonces la llamada al método

```
modificarArreglo( temperaturasPorHoraDia );
```

Matrices

Paso de arreglos a los métodos

Toda fila de la matriz “conoce” su propia cantidad de columnas (a través de su campo *length*) y la matriz la cantidad de filas (a través de su campo *length*) . Por ende, cuando pasamos a un método la referencia a un objeto matriz, no necesitamos pasar la cantidad de filas y columnas como argumentos adicionales.

Matrices

Paso de arreglos a los métodos

Para que un método reciba una referencia de una matriz a través de una llamada a un método, la lista de parámetros del método debe especificar un parámetro tipo matriz.

Matriz

Paso de arreglos a los métodos

Por ejemplo, el encabezado para el método modificarMatriz podría escribirse así:

```
public static void modificarArreglo(int [][] b){
//....
}
```

lo cual indica que modificarMatriz recibe la referencia de una matriz de enteros en el parámetro b.

Matriz

Paso de arreglos a los métodos

La llamada a este método pasa la referencia a la matriz *temperaturasPorHoraDia* , de manera que cuando el método llamado utiliza la variable b tipo matriz, hace referencia al mismo objeto matriz como temperaturasPorHora en el método que hizo la llamada.

Conclusiones

- En teoría de los grafos, a todo grafo etiquetado corresponde la matriz de adyacencia. Una matriz de permutación es una matriz que representa una permutación; matriz cuadrada cuyos coeficientes son 0 o 1, con un solo 1 en cada línea y cada columna. Estas matrices se utilizan en combinatorio.

Conclusiones

- En la teoría de grafos, se llama matriz de un grafo a la matriz que indica en la línea i y la columna j el número de aristas que enlazan el vértice i al vértice j. En un grafo no orientado, la matriz es simétrica.

Conclusiones

- Como todos los objetos, las matrices se pasan como argumentos a los métodos por referencia.
- Se pueden crear matrices anónimas (por ejemplo, crear una nueva ,matriz como argumento actual en la llamada a un método).

Conclusiones

- A veces el iterar o moverse dentro de una matriz de cierta manera puede contribuir de forma positiva o negativa en la búsqueda de la solución.

UNIVERSIDAD
DE MATANZAS

cosechando el saber

FIN