Matrices en Java

Profesor: Andrés Guzmán F

¿Qué son las matrices?

Son arreglos bidimensionales, donde cada uno de sus elementos es a su vez un arreglo en su segunda dimensión

Esto permite que no todos los elementos tengan el mismo tamaño

De esta forma es posible crear matrices recursivas y multi-dimensionales

¿Qué son las matrices?

Son arreglos bidimensionales, donde cada uno de sus elementos es a su vez un arreglo en su segunda dimensión

Esto permite que no todos los elementos tengan el mismo tamaño

De esta forma es posible crear matrices recursivas y multi-dimensionales

Comienzan en índice 0

	0	1	2	3	4	5
0	1	2	3	4	5	6
1	20	22	24	26	28	30
2	31	33	35	37	39	40
3	50	60	70	80	90	100

¿Qué son las matrices?

Son arreglos bidimensionales, donde cada uno de sus elementos es a su vez un arreglo en su segunda dimensión

Esto permite que no todos los elementos tengan el mismo tamaño

De esta forma es posible crear matrices recursivas y multi-dimensionales

Comienzan en índice 0

Total de columnas length = 6

	0	1	2	3	4	5
0	1	2	3	4	5	6
1	20	22	24	26	28	30
2	31	33	35	37	39	40
3	50	60	70	80	90	100

Total de filas length = 4

¿Qué son las matrices?

Son arreglos bidimensionales, donde cada uno de sus elementos es a su vez un arreglo en su segunda dimensión

Esto permite que no todos los elementos tengan el mismo tamaño

De esta forma es posible crear matrices recursivas y multi-dimensionales

Comienzan en índice 0

Total de columnas length = 6

	0	1	2	3	4	5
0	1	2	3	4	5	6
1	20	22	24	26	28	30
2	31	33	35	37	39	40
3	50	60	70	80	90	100

Total de filas length = 4

num 37 se encuentra en la posición [2][3]

Declaración e instanciación

La declaración de una matriz tiene dos partes: el tipo de datos del arreglo seguido de dobles corchetes y el nombre de la variable

```
int[][] numeros;
```

o bien

```
int numeros[][];
```

Al igual que el tipo de array incluye el tipo de dato de los elementos que va contener

Cuando se crea una matriz, se utiliza el operador new, más el tipo de los elementos, más el número de filas y columnas

```
int[][] numeros = new int[fila][columna];

// ejemplo
int[][] numeros = new int[2][3];
```

Tamaños de filas y columnas

- Como en los arreglos, podemos obtener el tamaño de la matriz con el atributo length, tanto para las filas y columnas:

```
int[][] numeros = new int[2][3];
System.out.println("numero de filas = " + numeros.length);
System.out.println("número de columnas = " + numeros[0].length);
```

Declaración e instanciación

- Pero, no solo podemos almacenar elementos del tipo primitivos, sino que también del tipo de referencia, objetos!

```
Producto[][] productos = new Producto[2][2];

String[][] nombres = new String[3][2];
```

Inicialización de elementos

- Asignamos elementos a la matriz indicando la llave o índice de la fila y de la columna

```
int[][] numeros = new int[2][4];
numeros[0][0] = 1;
numeros[0][1] = 2;
numeros[0][2] = 3;
numeros[0][3] = 4;
```

Inicialización de elementos

- Asignamos elementos indicando el índice de la fila y columna

```
int[][] numeros = new int[2][4];
numeros[0][0] = 1;
numeros[0][1] = 2;
numeros[0][2] = 3;
numeros[0][3] = 4;

numeros[1][0] = 11;
numeros[1][1] = 12;
numeros[1][2] = 13;
numeros[1][3] = 14;
```

Inicialización de elementos

- Asignamos elementos indicando el índice de la fila y columna

```
int[][] numeros = new int[2][4];
numeros[0][0] = 1;
numeros[0][1] = 2;
numeros[0][2] = 3;
numeros[0][3] = 4;

numeros[1][0] = 11;
numeros[1][1] = 12;
numeros[1][2] = 13;
numeros[1][3] = 14;
```

	0	1	2	3
0	1	2	3	4
1	11	12	13	14

Inicialización de elementos

- Asignamos elementos indicando el índice de la fila y columna

```
Producto[][] productos = new Producto[2][2];

productos[0][0] = new Producto("Mesa Comedor");
productos[0][1] = new Producto("TV Sony LED 55");
productos[1][0] = new Producto("Bicicleta Oxford");
productos[1][1] = new Producto("Bicicleta Estática Gimnasio");
```

Inicialización de elementos

- Asignamos elementos indicando el índice de la fila y columna

```
Producto[][] productos = new Producto[2][2];

productos[0][0] = new Producto("Mesa Comedor");
productos[0][1] = new Producto("TV Sony LED 55");
productos[1][0] = new Producto("Bicicleta Oxford");
productos[1][1] = new Producto("Bicicleta Estática Gimnasio");
```

	0	1
0	Mesa Comedor	TV Sony LED 55
1	Bicicleta Oxford	Bicicleta Estática Gimnasio

Inicialización de elementos

- Asignamos elementos indicando el índice de la fila y columna

```
String[][] nombres = new String[3][2];
nombres[0][0] = "Pepe";
nombres[0][1] = "Maria";

nombres[1][0] = "Pato";
nombres[1][1] = "Bea";

nombres[2][0] = "Lucas";
nombres[2][1] = "Luci";
```

Inicialización de elementos

- Asignamos elementos indicando el índice de la fila y columna

```
String[][] nombres = new String[3][2];
nombres[0][0] = "Pepe";
nombres[0][1] = "Maria";

nombres[1][0] = "Pato";
nombres[1][1] = "Bea";

nombres[2][0] = "Lucas";
nombres[2][1] = "Luci";
```

	0	1
0	Pepe	Maria
1	Pato	Bea
2	Lucas	Luci

Obtener elementos

- Accedemos a los elementos de la matriz mediante índice o llaves de la fila y columna

```
int num1 = numeros[0][0];
int num2 = numeros[0][1];
int num3 = numeros[0][2];
int num4 = numeros[0][3];
```

Obtener elementos

- Accedemos a los elementos de la matriz mediante índice de la fila y columna

```
int num1 = numeros[0][0];
int num2 = numeros[0][1];
int num3 = numeros[0][2];
int num4 = numeros[0][3];

Producto mesa = productos[0][0];
Producto tvSony = productos[0][1];
Producto bici = productos[1][0];
```

Obtener elementos

- Accedemos a los elementos de la matriz mediante índice de la fila y columna

```
int num1 = numeros[0][0];
int num2 = numeros[0][1];
int num3 = numeros[0][2];
int num4 = numeros[0][3];

Producto mesa = productos[0][0];
Producto tvSony = productos[0][1];
Producto bici = productos[1][0];

String nombre1 = nombres[0][0];
String nombre2 = nombres[0][1];
String nombre3 = nombres[1][0];
String nombre4 = nombres[1][1];
String nombre5 = nombres[2][0];
String nombre5 = nombres[2][1];
```

Declaración, instanciación e inicialización de una matriz

- Se utiliza cuando conocemos los elementos y el tamaño de la matriz

```
int[][] numeros = { {1, 2, 3, 4}, {11, 12, 13, 14} };
```

Declaración, instanciación e inicialización de una matriz

- Se utiliza cuando conocemos los elementos y el tamaño

```
int[][] numeros = {  {1, 2, 3, 4}, {11, 12, 13, 14}  };

Producto[][] productos = {
            { new Producto("Mesa Comedor"), new Producto("TV Sony LED 55") },
            { new Producto("Bicicleta Oxford"), new Producto("Bicicleta Estática Gimnasio") }
};

String[][] nombres = {  {"Pepe", "Maria"}, {"Pato", "Bea"}, {"Lucas", "Luci"}  };
```

Recorrer una matriz usando for

```
String[][] nombres = new String[3][2];
  nombres[0][0] = "Pepe";
  nombres[0][1] = "Maria";

  nombres[1][0] = "Pato";
  nombres[1][1] = "Bea";

  nombres[2][0] = "Lucas";
  nombres[2][1] = "Luci";

  for(int i = 0; i < nombres.length; i++) {

     for(int j = 0; j < nombres[i].length; j++) {
        System.out.println("nombre = " + nombres[i][j]);
     }

  }
```