PROGRAMACIÓN

UT3: Arrays multidimensionales (IIII)

1. ¿Qué es un array bidimensional?

Podemos definir los arrays bidimensionales como tablas, con n filas y m columnas.

¿Por qué son necesarios?

Normalmente nos vamos a encontrar con la necesidad de almacenar más de un valor, como si fuese una lista o una tabla, por ejemplo, el nombre de los alumnos de una clase, las calificaciones de una asignatura de todos alumno, las calificaciones de muchos alumnos de una misma asignatura, representar el mapa de un juego, …

Con el uso de arrays bidimensionales solucionamos ese problema, ya que nos permiten almacenar todos estos valores en una sola variable como si de una tabla se tratase.

2. Estructura de un array

En estos arrays encontramos tres atributos:

- El índice de fila; nos indica la posición del array en la fila.
- El índice de columna; nos indica la posición del array en la columna.
- El elemento; nos indica qué contiene (el valor) el array en una determinada posición.

	0	**1**	**2**	**3**	**4**
0	10	9	7	5	9
1	8	7	9	4	5
2	8	7	4	10	4

ÍNDICE COLUMNA
ELEMENTO
ÍNDICE FILA

NOTA:

¡La primera posición de un array bidimensional siempre es el índice (0, 0)!

Los arrays pueden ser de cualquier tipo, pero TODOS los elementos deben ser del mismo.

Es decir, si un array es de enteros, todos los elementos deben ser enteros.

3. Declaración e inicialización de un array bidimensional

DECLARACIÓN

Para declarar un array en java seguiremos la siguiente estructura, pero debemos tener en cuenta que los arrays son “listas” estáticas, por lo que **siempre debemos indicarles cual va a ser su longitud al crearlos.**

NO SE PUEDEN AÑADIR MÁS ELEMENTOS DE LOS QUE TENGA EL ARRAY.

<tipo_datos> [ ] [ ] <nombre_variable> = new <tipo_datos> [<num_filas>] [<num_columnas>]

Con [ ] [ ] (los corchetes) indicamos que la variable va a ser un array bidimensional.

Ejemplo:

String [ ] [ ] arrray_Strings = new String [10] [4]; // Array de 10 filas y 4 columnas

int [ ] [ ] array_Enteros = new int [5] [2];

INICIALIZACIÓN

Al crear el array, sino le asignamos valores se inicializa a unos valores predeterminados en función del tipo del array.

- Variables numéricas a **cero**.
- Variables de caracteres a **\u0000**.
- Variables boleanas a **false**.
- Objetos a **null**.

Tenemos dos formas de inicializar un array, ir uno a uno dando valor a sus elementos, o inicializarlos todos a la vez.

Ejemplo:

String [ ] arrray_Strings = new String [12] [2]

array_Strings[0] [0] = “Javier”;

array_Strings [0] [1] = “García”;

array_Strings[1] [0] = “Marta”;

…

array_Strings[11] [1] = “Zambrano”;

Ejemplo2: // Declaramos el array de golpe:

int [ ] [ ] array_Enteros = new int [3] [2];

array_Enteros = { {4, 32} , {-3, 1} , {4, 3} };

NOTA

Si queremos declarar arrays de mas de 2 dimensiones, simplemente añadiremos un corchete más, y vamos añadiendo dimensiones.

Por ejemplo un array tridimensional se declararía:

int [ ] [ ] [ ] array_cuboRubik = new int [3] [3] [3];

4. Uso de un array

Podemos guardar el valor de cada posición del array en una variable para su posterior uso.

Del mismo modo que le damos valor, lo solicitamos:

Ejemplo:

```
int num = array_Enteros [2] [1];
```

Otra función útil de los arrays es conocer por código su longitud.

¡En este caso, la longitud nos indicará el NÚMERO DE FILAS!

Esto es posible usando "**length**"

Si escribimos

```
int num_filas = array_Enteros.length;
```

Si queremos conocer el número de columnas de este array, le preguntaremos lo siguiente:

```
int num_columnas = array_Enteros[0].length;
```

Donde el 0 representa la primera fila, y nos devolverá la longitud (numero de columnas) de esa fila!

Esto es muy útil cuando queremos recorrer un array entero, con el uso del bucle for. Aunque si tenemos 2 dimensiones, tendremos que encadenar 2 bucles. (Si tuviésemos n dimensiones, tendremos que encadenar n bucles for)

Ejemplo:

RECORRIDO POR FILAS!

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

RECORRIDO POR COLUMNAS! (Tenemos que tener en una variable el número de columnas)

```
for (int i = 0 ; i < numero_columnas; i++){
        for (int j = 0 ; j < array.length ; j++){
            System.out.println( array [ j ] [ i ]);
        }
}
```