

Las matrices consisten en crear *arrays* de *arrays*, de la siguiente forma:

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

Me gusta ver esto como dimensiones, si queremos representar 0 dimensiones, es decir, un punto o un dato en este caso, ponemos 0 corchetes, si queremos hacer un *array*, una línea de datos, ponemos 1 corchetes, y si queremos hacer una matriz, que viene a ser un plano (2 dimensiones), ponemos 2 corchetes, para 3 dimensiones → 3 corchetes, etc.

Ahora bien, como veis he creado una matriz de 3x3, pero no tienen que coincidir tan exactos los números, por ejemplo:

```
static int[][] matriz = {{1},{2,3,4},{5,6}};
```

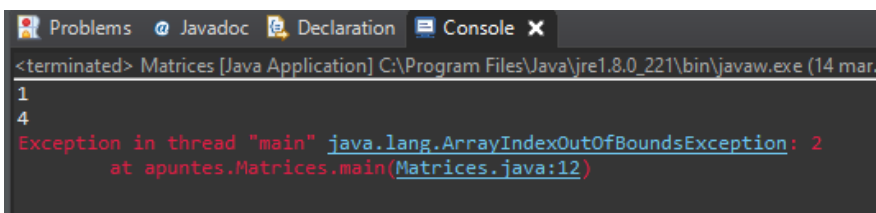
Ahora he creado una matriz de 3 filas, pero la primera tiene una sola columna, la segunda 3 y la tercera 2, enumerándolas, podemos hacer que las columnas tengan el tamaño que queramos, pero claro, si no tienen todas las mismas columnas, no serían matrices como tal, creo vaya, no soy matemático, el caso es que por enumeración podemos hacer esto, pero si le indicamos el tamaño siempre van a ser matrices, si o si:

```
static int[][] matriz = new int [3][4];
```

He creado una matriz de 3 filas y 4 columnas, que como ya vimos, todos los elementos son 0, voy a ponerlos unos ejemplos con la “matriz” de la segunda foto, la que tiene columnas de diferentes tamaños:

```
System.out.println(matriz[0][0]);  
System.out.println(matriz[1][2]);  
System.out.println(matriz[2][2]);
```

>>>



Como veis, el elemento [0][0] es el 1, el [1][2] es el 4, y al tratar de usar el elemento [2][2] se nos va de rango, porque el tercer *array* de nuestra matriz, solo tiene dos elementos (el 0 y el 1), recordad que empezamos en el 0, veamos otro ejemplo:

```
System.out.println(matriz[0]); // {1}  
System.out.println(matriz[1]); // {2,3,4}  
System.out.println(matriz[2]); // {5,6}
```

>>>

```
[I@15db9742  
[I@6d06d69c  
[I@7852e922
```

Los elementos de nuestra matriz son *arrays*, por lo que al tratar de imprimirlos nos devuelven la dirección de memoria.

Vamos a ver cómo usar la longitud:

```
System.out.println(matriz.length);  
System.out.println(matriz[0].length); // {1}  
System.out.println(matriz[1].length); // {2,3,4}  
System.out.println(matriz[2].length); // {5,6}
```

>>>

```
3  
1  
3  
2
```

Bastante fácil creo yo, la primera nos muestra el numero de filas de la matriz, y en los otros la cantidad de columnas de la fila que sea, vamos a crear una matriz de *Strings*, donde cada elemento muestre su posición:

```
static String[][] matriz = new String [4][4];  
  
public static void main (String[] args) {  
    for (int f = 0; f < matriz.length; f++) {  
        for (int c = 0; c < matriz[f].length; c++) {  
            matriz[f][c] = f + " " + c;  
        }  
    }  
}
```

La variable *f* recorre todas las filas, es decir, va desde 0, hasta la longitud de la matriz, sin llegar a ésta, ya que como sabemos, un *array* de tamaño 4 tiene el intervalo [0,3]. La variable *c*, en vez de llegar a *matriz.length* llega hasta *matriz[f].length*, para que recorra todas las columnas de esa fila en específico, esto solo se hace si las filas tienen diferente numero de columnas, como la matriz de la primera foto de ésta página, pero en este caso en vez de poner *matriz[f].length* podríamos poner *matriz[0].length*, ya que todas las filas tienen el mismo número de columnas, y si fuese una matriz **cuadrada**, tendría el mismo número de filas que de columnas, por lo que con poner *matriz.length* ya bastaría, hemos de tener cuidado a la hora de poner hasta donde va a llegar nuestro bucle recorriendo *arrays* o matrices, pero puedo asegurar que la forma en la que lo hago arriba funciona para cualquier matriz, de cualquier dimensión.

Vamos a crear una función que nos convierta una matriz en un solo *String*; NOTA → No hace falta que sea una matriz de *Strings* para poder convertirlo, por si alguien se lía con esto, recordad que cualquier tipo de dato **primitivo** puede convertirse a *String*:

```
public static String aString(String[][] matriz) {  
    String res = "";  
    for (int f = 0; f < matriz.length; f++) {  
        for (int c = 0; c < matriz[f].length; c++) {  
            res += matriz[f][c] + " ";  
        }  
        res += "\n";  
    }  
    return res;  
}  
  
public static void main (String[] args) {  
    for (int f = 0; f < matriz.length; f++) {  
        for (int c = 0; c < matriz[f].length; c++) {  
            matriz[f][c] = f + "" + c;  
        }  
    }  
    System.out.println(aString(matriz));  
}
```

>>>

```
00 01 02 03  
10 11 12 13  
20 21 22 23  
30 31 32 33
```

**IMPORTANTE:** No llameís a una función *toString* **NUNCA**, esto es algo que entenderéis al llegar a “Clases y Objetos”, demomento dejemoslo que no tienen que tener ese nombre, aunque al ser del tipo *static* no pasa nada, es mejor no llamarlas así, dicho esto, continuemos;

Como veis en la función *aString* creo una variables *res* en la que iré introduciendo el resultado, voy recorriendo cada elemento de la matriz, se lo sumo junto a un espacio, y cada vez que cambio de fila, es decir, cuando acaba el bucle de la variable *c*, y pasa a incrementarse el de la variable *f*, metemos un salto de línea, metiendole un “\n”, no tienen nada más las matrices, como siempre tendréis unos ejercicios en mi carpeta de Drive.