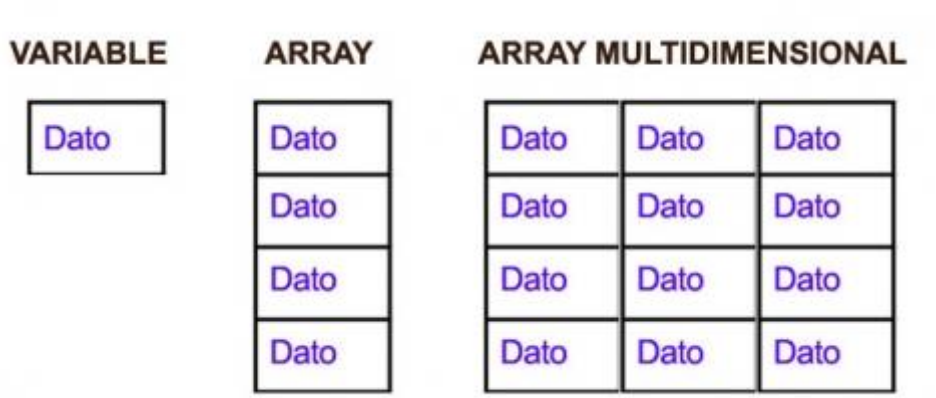
Arrays multidimensionales

Las matrices multidimensionales son matrices de matrices o **arrays de arrays**, donde cada elemento del array contiene la referencia de otro array. Se crea una matriz multidimensional al agregar un conjunto de corchetes ([]) por dimensión.

Una imagen ilustrativa para mejor comprensión:



Ejemplos:

```
int[][] intArray = new int; //un array 2D o matrix
int[][][] intArray = new int; //una array 3D
```

Ejemplo:

```
class multiDimensional
{
   public static void main(String args[])
   {
      // declarar e inicializar array 2D
     // dos filas, tres columnas
      int arr [ ][ ]  arr = new int[2][3];
      arr[][] = { {2,7,9},{3,6,1},{7,4,2} };

      // imprimir array 2D
      for (int i=0; i< 3 ; i++)
      {
         for (int j=0; j < 3 ; j++)
            System.out.print(arr[i][j] + " ");
```

```
        System.out.println();
    }
  }
}
```

Salida:

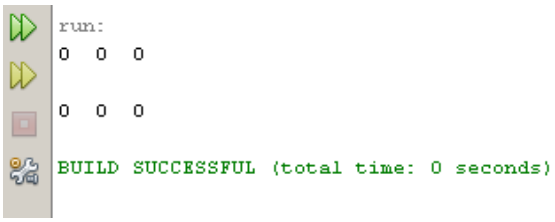
```
2 7 9
3 6 1
7 4 2
```

Fuente: https://javadesdecero.es/arrays/unidimensionales-multidimensionales/#3_Arrays_multidimensionales

Ejemplo 2:

```
1  package aulafacil;
2
3  public class AulaFacil {
4
5      public static void main(String[] args) {
6
7          int[][] numeros; //Un array multidimensional de números enteros
8          int i, j; // Para recorrer el Array: i = filas, j = columnas
9
10         //Este Array tiene 2 filas y 3 columnas
11         numeros = new int[2][3]; //Le doy el nº de filas y columnas
12
13         for (i = 0; i < numeros.length; i++) //Recorre filas
14         {
15             for (j = 0; j < numeros[i].length; j++) //De cada fila, recorre todas
16                                                     //sus columnas
17             {
18                 numeros[i][j] = 0;
19             }
20         }
21
22     }
23 }
24
25                                                     AulaFacil.com
```

```
22         for (i = 0; i < numeros.length; i++) //Recorre filas
23         {
24             for (j = 0; j < numeros[i].length; j++) //De cada fila, recorre todas
25                                                     //sus columnas
26             {
27                 //Mostrando elementos de una fila
28                 System.out.print(numeros[i][j] + "  ");
29             }
30             // \n es un salto de línea...Para mostrar la otra fila debajo
31             System.out.println("\n");
32         }
```



El tipo de dato puede ser cualquiera de los admitidos por Java y que ya ha sido explicado. Ejemplos de declaración e inicialización con valores por defecto de arrays, usando los distintos tipos de datos Java, serían:

- byte[][] edad = new byte[4][3];
- short][] edad = new short[4][3];
- int[][] edad = new int[4][3];
- long[][] edad = new long[4][3];
- float[][] estatura = new float[3][2];
- double[][] estatura = new double[3][2];
- boolean[][] estado = new boolean[5][4];
- char[][] sexo = new char[2][1];
- String[][] nombre = new String[2][1];

La declaración de una matriz tradicional de m x n elementos podría ser:

/* Ejemplo declaración */

int[][] ***matriz*** = new int[3][2];

O alternativamente

int[][] ***matriz***;

matriz = new int[3][2];

El número de elementos sería: 3 x 2 = 6, dónde 3 es el número de filas y 2 es el número de columnas.

Ahora procedemos a cargar la matriz con valores:

matriz[0][0] = 1;

matriz[0][1] = 2;

matriz[1][0] = 3;

matriz[1][1] = 4;

matriz[2][0] = 5;

matriz[2][1] = 6;

Hay que recordar que los elementos empiezan a numerarse por 0. Así, la esquina superior izquierda de la matriz será el elemento [0][0] y la esquina inferior derecha será el [2][1]. Hay que prestar atención a esto porque en otros lenguajes de programación la numeración puede empezar por 1 en vez de por 0.

También se pueden cargar directamente los elementos, durante la declaración de la matriz de la siguiente manera:

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

donde {1,2} corresponde a la fila 1, {3,4} a la fila 2 y {5,6} a la fila 3, y los números separados por coma dentro de cada fila, corresponden a las columnas. En este caso, los números (1, 3, 5) de cada una de las filas corresponden a la primera columna y los números (2, 4, 6) atañen a la segunda columna.

Para obtener el número de filas de la matriz, podemos recurrir a la propiedad "length" de los arrays, de la siguiente manera:

```
int filas = matriz.length;
```

Para el caso del número de columnas sería de la siguiente forma:

```
int columnas = matriz[0].length;
```

También Java nos permite la posibilidad de clonar una matriz, es decir, crear una matriz nueva a partir de otra matriz, siguiendo esta sintaxis:

String[][] ***nuevaMatriz*** = ***matriz***.clone();

donde clone() es un método especial, que permite la clonación de arrays de cualquier dimensión en Java. De esta manera "nuevaMatriz" y "matriz" son 2 matrices distintas, pero con los mismos valores.

EJERCICIO RESUELTO

Vamos a plantear y resolver un ejercicio: queremos almacenar en una matriz el número de alumnos con el que cuenta una academia, ordenados en función del nivel y del idioma que se estudia. Tendremos 3 filas que representarán al Nivel básico, medio y de perfeccionamiento y 4 columnas en las que figurarán los idiomas (0 = Inglés, 1 = Francés, 2 = Alemán y 3 = Ruso). Se pide realizar la declaración de la matriz y asignarle unos valores de ejemplo a cada elemento.

SOLUCIÓN

La declaración de la matriz sería:

```
/* Ejemplo declaración*/
int[][]  alumnosfxniveleidioma= new int[3][4];
```

Podríamos asignar contenidos de la siguiente manera:

alumnosfxniveleidioma[0][0] = 7; alumnosfxniveleidioma[0][1] = 14;

alumnosfxniveleidioma[0][2]= 8; alumnosfxniveleidioma[0][3] = 3;

alumnosfxniveleidioma[1][0] = 6; alumnosfxniveleidioma[1][1]= 19;

alumnosfxniveleidioma[1][2] = 7; alumnosfxniveleidioma[1][3] = 2

alumnosfxniveleidioma[2][0] = 3; alumnosfxniveleidioma[2][1]= 13;

alumnosfxniveleidioma[2][2] = 4; alumnosfxniveleidioma[2][3] = 1

También, podríamos asignar contenido de esta otra forma, como ya se ha explicado anteriormente:

int[][] alumnosfxniveleidioma = {{7,14,8,3},{6,19,7,2},{3,13,4,1}};

La representación gráfica que podríamos asociar a esta asignación de datos sería esta matriz:

$$\begin{pmatrix} 7 & 14 & 8 & 3 \\ 6 & 19 & 7 & 2 \\ 3 & 13 & 4 & 1 \end{pmatrix}$$

La organización de la información en matrices nos generará importantes ventajas a la hora del tratamiento de datos en nuestros programas.

Para terminar en cuanto a multidimensionalidad, veamos casos de declaraciones con más de dos dimensiones. Para ello supongamos que estamos realizando un "conteo de coches", es decir, que estamos contando los coches que pasan por un determinado lugar en un periodo de tiempo que puede ser un día, varios días, varios meses, etc. La forma de declarar esos arrays podría ser la siguiente:

Duración del conteo	**Tipo de array**	**Declaración con Java (nc es Número de coches)**
Un día	Array de una dimensión (hora)	int[] nc = new int[24];
Varios días	Array de dos dimensiones (hora y día)	int[][] nc = new int[24][31];
Varios meses	Array de tres dimensiones (hora, día y mes)	int[][][] nc = new int[24][31][12];
Varios años	Array de cuatro dimensiones (hora, día, mes y año)	Int[][][][] nc = new int[24][31][12][2999];

Varios siglos	Array de cinco dimensiones (hora, día, mes, año y siglo)	Int[][][][][] nc = new int[24][31][12][2999][21];

Veamos lo que sería un ejemplo de programa con array multidimensional, usando un tipo String.

```
/* Ejercicio Array multidimensional */
public class MatrizAlumnos {
  public static void main(String arg[]) {
        String[ ][ ]  nombreAlumno = new String[5][25];
        nombreAlumno[2][23] = "Pedro Hernández González";
        System.out.println("El alumno número 24 del curso tercero se llama "+nombreAlumno[2][23]);
  }
}
```

El resultado del programa es la aparición del mensaje "El alumno número 24 del curso tercero se llama Pedro Hernández González.

En este ejemplo, [5] representa a los cursos. Hablamos de 5 cursos que son identificados con 0, 1, 2, 3, 4, por lo que [2] hace mención al tercer curso; lo mismo podemos decir de [23], que corresponde al alumno número 24. Hay que recordar que siempre en Java tenemos que contar el cero, ya que si no lo hacemos podemos cometer errores.

EJERCICIO

Crea un programa que pida por pantalla cuatro países y a continuación tres ciudades de cada uno de estos países. Los nombres de ciudades deben almacenarse en un array multidimensional cuyo primer índice sea el número asignado a cada país y el segundo índice el número asignado a cada ciudad.

Ejemplo de resultados que debe mostrar el programa:

País: Argentina Ciudades: Buenos Aires Córdoba La Plata

País: España Ciudades: Madrid Lugo Sevilla

País: Francia Ciudades: Paris Niza Lyon

País: Italia Ciudades: Roma Nápoles Sicilia

Fuente: https://www.aprenderaprogramar.com/