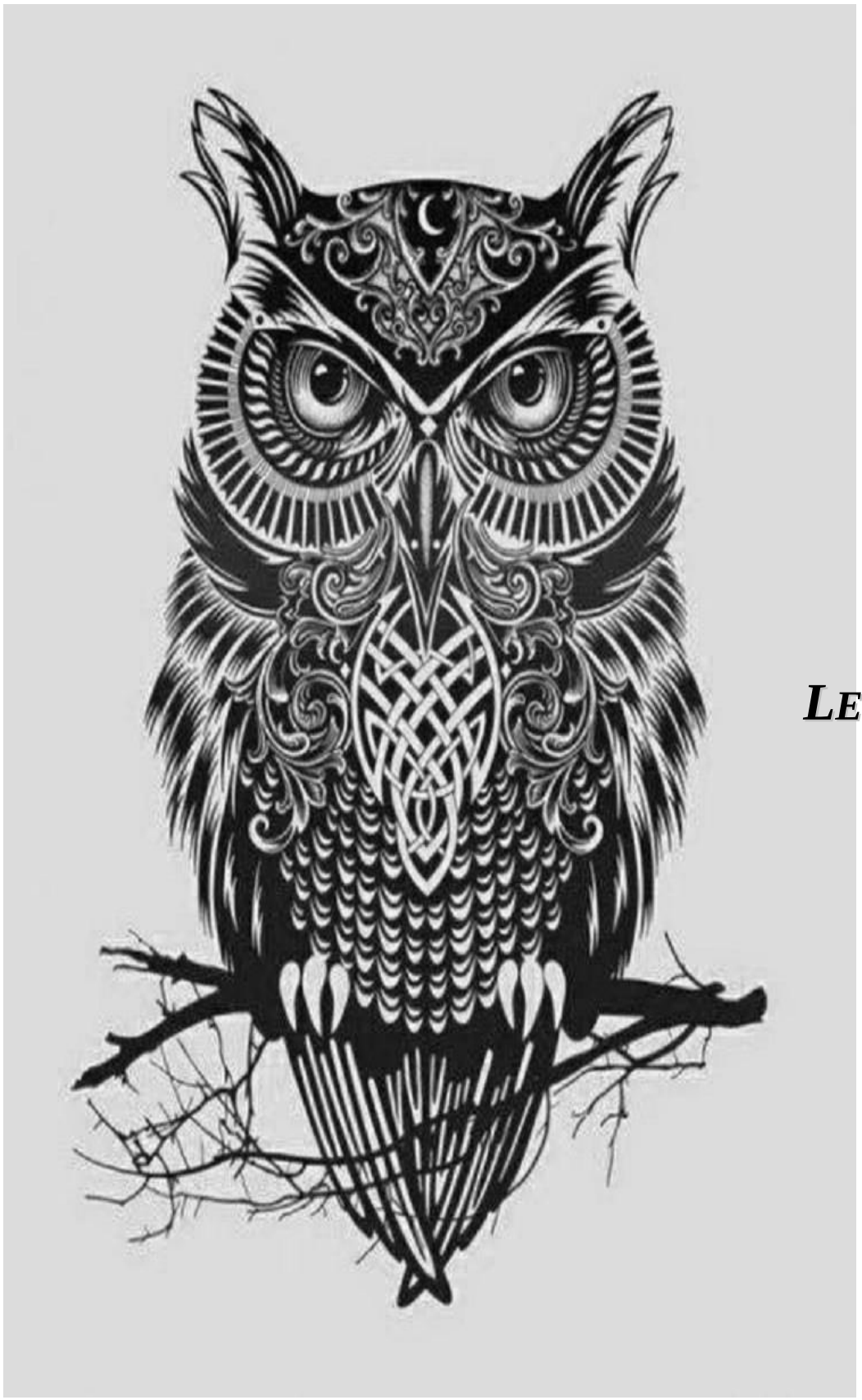


ARREGLOS (ARRAYS) EN EL LENGUAJE JAVA

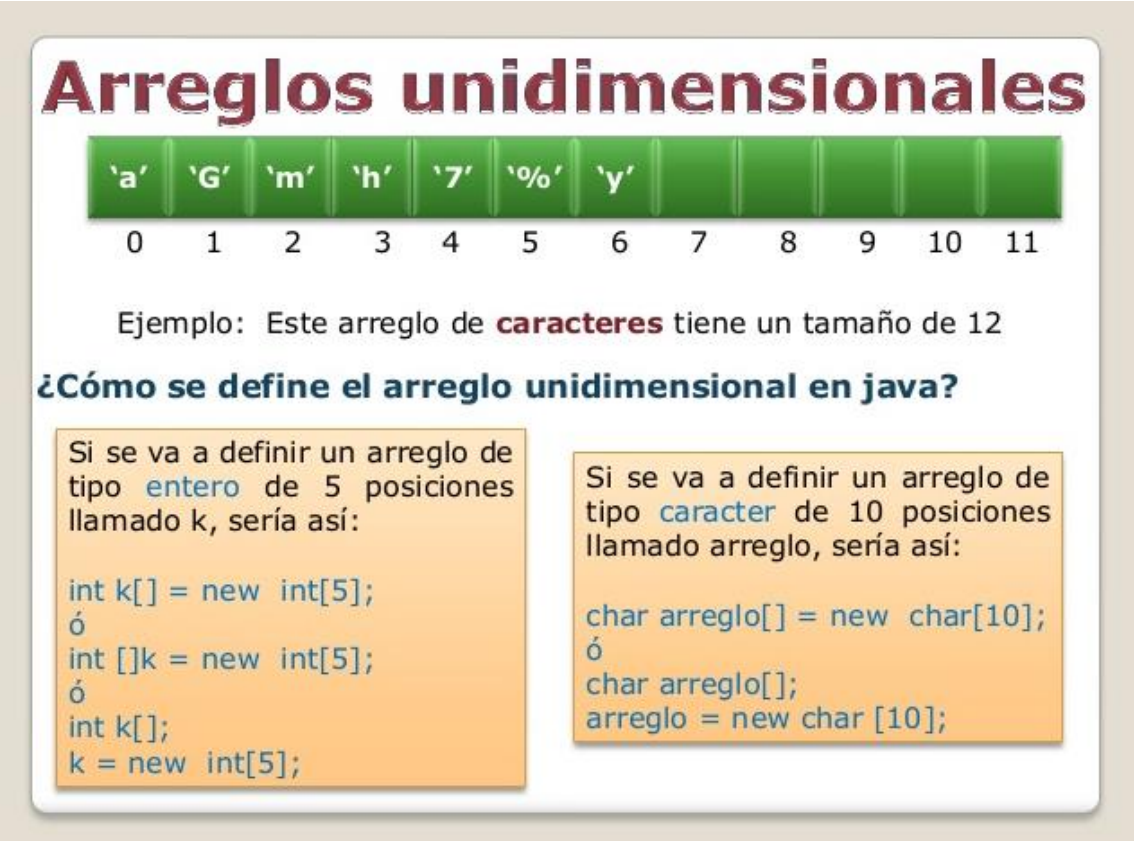
LUIS ERNESTO
RUBIO TORRES

NOVIEMBRE 2018

Arreglos

Un array (arreglo) en Java es una estructura de datos que nos permite almacenar un conjunto de datos de un mismo tipo.

El tamaño de los arrays se declara en un primer momento y no puede cambiar luego durante la ejecución del programa, como sí puede hacerse en otros lenguajes. Veremos ahora cómo declarar arrays estáticos de una dimensión.



http://image.slidesharecdn.com/16arreglosunidimensionalesjava-130611131106-phpapp02/95/16-curso-de-poo-en-java-arreglos-unidimensionales-4-638.jpg?cb=1394801484

La sintaxis para declarar e inicializar un array será:

```
Tipo_de_variable[ ]  Nombre_del_array = new  Tipo_de_variable[dimensión];
```

También podemos alternativamente usar esta declaración:

```
Tipo_de_variable[ ] Nombre_del_array;

Nombre_del_array = new Tipo_de_variable[dimensión];
```

El tipo de variable puede ser cualquiera de los admitidos por Java y que ya hemos explicado.

Ejemplos de declaración e inicialización con valores por defecto de arrays usando todos los tipos de variables Java, serían:

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

Aclarar que los valores por defecto son los siguientes:

a) Para números el valor cero "0".

b) Para cadenas y letras el valor vacío.

c) Para booleanos el valor false.

En caso de que queramos inicializarlos con valores propios, haremos esto:

Para números enteros

```
int[ ] edad = {45, 23, 11, 9};  //Array de 4 elementos
```

De la misma forma procederíamos para los otros tipos de enteros: byte, short, long.

Para números reales

```
double[ ] estatura = {1.73, 1.67, 1.56}; //Array de 3 elementos
```

De la misma forma procederíamos para el tipo float, pero teniendo en cuenta que los números deberán llevar al final la letra "f" o "F". Por ejemplo 1.73f o 1.73F.

Para cadenas

```
String[ ] nombre = {"María", "Gerson"};   //Array de 2 elementos
```

Para caracterers

char[ ] sexo = {'m', 'f'}; //Array de 2 elementos

Para booleanos

boolean[ ] = {true,false}; //Array de 2 elementos

Cuando creamos un array de nombre "a" y de dimensión "n" (int[ ] a = new int[n]) estamos creando n variables que son a[0], a[1], a[2], ..., a[n-1].

Los arrays se numeran desde el elemento cero, que sería el primer elemento, hasta el n-1 que sería el último elemento.

Es decir, si tenemos un array de 5 elementos, el primer elemento sería el cero y el último elemento sería el 4.

Esto conviene tenerlo en cuenta porque puede dar lugar a alguna confusión. Disponer de un valor con índice cero puede ser de utilidad en situaciones como considerar cada variable asociada a una hora del día, empezando a contar desde la hora cero hasta la 23 (total de 24 horas), cosa que es habitual en algunos países. En lugar de 1, 2, 3, ..., 24 estaríamos usando 0, 1, 2, ..., 23.

Vamos a trabajarlo sobre el ordenador en un programa y ver qué pasaría si declaramos un array de tamaño "n" y vamos asignando un valor a cada variable del array desde la posición cero hasta la posición "n-1". Adicionalmente vamos a comprobar qué ocurre si tratamos de asignarle valor a la variable de posición "n".

El código fuente del programa es el siguiente:

```
  /* Ejemplo uso Arrays  */
public class ArrayDeNombres {
    public static void main(String arg[ ]) {
        String[ ] nombre = new String[4];
        nombre[0] = "Luis";
        nombre[1] = "María";
        nombre[2] = "Carlos";
        nombre[3] = "Jose";
  }
}
```

Arreglos Multidimensionales

En Java es posible crear arrays con más de una dimensión, pasando de la idea de lista, vector o matriz de una sola fila a la idea de matriz de m x n elementos, estructuras tridimensionales, tetradimensionales, etc.

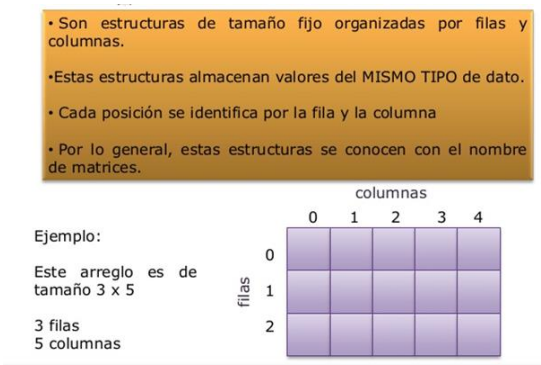
La sintaxis será:

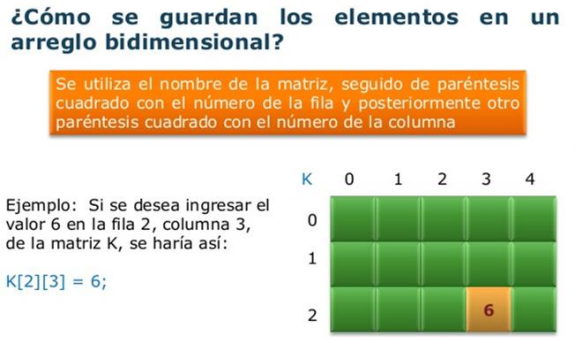
Tipo_de_variable[ ][ ]… [ ] Nombre_del_array = new Tipo_de_variable[dimensión1][dimensión2]…[dimensiónN];

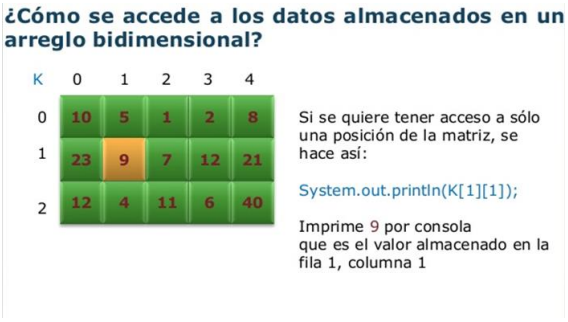
También podemos alternativamente usar esta declaración:

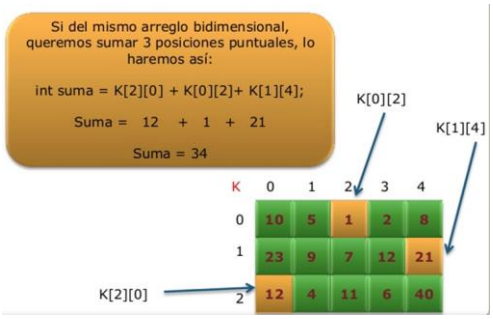
Tipo_de_variable[ ][ ] … [ ] Nombre_del_array;

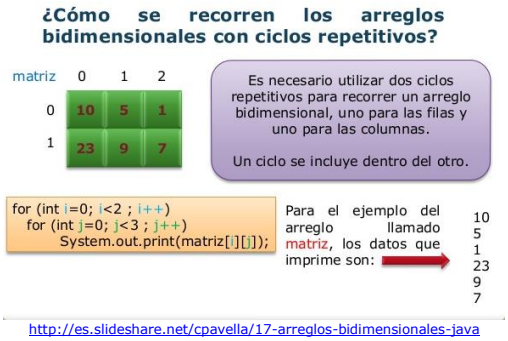
Nombre_del_array = new Tipo_de_variable[dimensión1][dimensión2]…[dimensiónN];











http://es.slideshare.net/cpavella/17-arreglos-bidimensionales-java

El tipo de variable 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 variables 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. Hablaremos del método clone más adelante.

Fuente:

http://aprenderaprogramar.com/index.php?option=com_content&view=article&id=168:repaso-arrays-o-arreglos-unidimensionales-en-java-tipos-de-inicializacion-ejemplos-de-codigo-cu00902c&catid=58:curso-lenguaje-programacion-java-nivel-avanzado-i&Itemid=180

http://aprenderaprogramar.com/index.php?option=com_content&view=article&id=233:arrays-arreglos-multidimensionales-en-java-declaracion-y-uso-ejemplos-y-ejercicios-resueltos-cu00904c&catid=58:curso-lenguaje-programacion-java-nivel-avanzado-i&Itemid=180