APRENDERAPROGRAMAR.COM

# REPASO ARRAYS O ARREGLOS UNIDIMENSIONALES EN JAVA. EJEMPLOS DE CÓDIGO. (CU00903C)

Sección: Cursos
Categoría: Lenguaje de programación Java nivel avanzado I

Fecha revisión: 2039

**Resumen:** Entrega nº3 del curso "Lenguaje de programación Java Nivel Avanzado I".

Autor: Manuel Sierra y José Luis Cuenca

**VARIABLES CON ÍNDICE O LOCALIZADOR. ARRAYS.**

Vamos a repasar algunos conceptos que ya deberíamos conocer. Un array 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.

**ARRAYS UNIDIMENSIONALES**

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
```

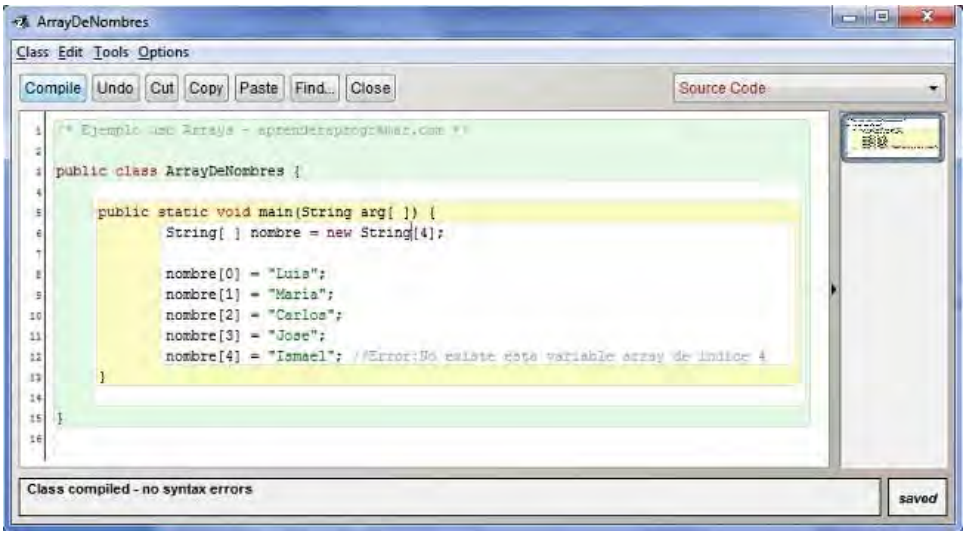
Cuando creamos un array de nombre “a” y de dimensión “n” (int[ ] a = new int[n]) estamos creando *n* variables que son *a[0], a[1], a[2], ..., a[n-1]*. Los arrays se numeran desde el elemento cero, que sería el primer elemento, hasta el n-1 que sería el último elemento. Es decir, si tenemos un array 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 (escríbelo en BlueJ) es el siguiente:

```
/* Ejemplo uso Arrays – aprenderaprogramar.com */

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";
        nombre[4] = "Ismael"; //Error:No existe esta variable array de índice 4
    }

}
```

Procedemos a la compilación del programa pulsando en el botón de Compile y después a su ejecución:




Obtenemos una respuesta (interpretación) que es la siguiente:

Una vez escrito el programa, cuando compilamos observamos que no hay ningún error de sintaxis, es decir, no faltan puntos y comas, las llaves están bien equilibradas, las palabras reservadas del lenguaje son las correctas, etc. Por tanto, aparentemente el programa debería ejecutarse y darnos un resultado. Sin embargo cuando cuando interpretamos el programa observamos un error "ArrayIndexOutBoundsException". Este mensaje de error significa "desbordamiento de array", es decir, que tratamos de acceder a una posición del array que no existe. En nuestro caso, hemos tratado de acceder a la posición 4, que no está definida (sólo se ha definido hasta la posición 3). En esto Java es muy estricto y no admite este tipo de error lógico del programador. Siguiendo el análisis del error que nos aparece en la ventana leemos lo siguiente "ArrayDeNombres.java:12", que nos está indicando que el error está en el archivo que contiene el programa de nombre "ArrayDeNombres.java", y más concretamente en la línea 12.

Ahora ampliaremos conocimientos con respecto a la instrucción **public static void main(String arg[ ])**. Esta instrucción es un estándar en Java, es decir, es la forma habitual de encabezar un programa que se ejecuta en consola. Ahora trataremos de entender un poco mejor el significado de este encabezado. Vamos a analizar el significado de (String arg[ ]). En esta instrucción el método main admite que se le pase un "array de Strings" cuando se proceda a la ejecución del programa si así lo deseamos. String arg[] significa "un array de Strings" cuyo nombre es arg. Dicho de otra manera, al ejecutar el programa podemos indicar "parámetros de entrada" de forma opcional. Si especificamos parámetros de entrada, éstos parámetros quedan asignados al array arg tomando el orden arg[0], arg[1].

Lo dicho podemos aclararlo mejor con el siguiente ejemplo:

El código fuente del programa es el siguiente. Escríbelo en el editor de BlueJ:

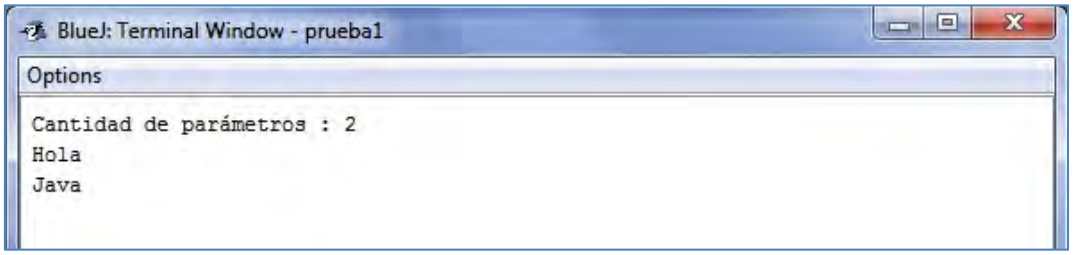
```
/* Ejemplo uso parámetros de entrada – aprenderaprogramar.com */

public class ParametrosDeEntrada {

    public static void main(String arg[ ]) {
        System.out.println("Cantidad de parámetros : " + arg.length);
        System.out.println(arg[0]);
        System.out.println(arg[1]);
    }

}
```

Damos por entendido que ya hemos compilado el programa y no hay errores de sintaxis. Nos centramos por tanto en la ejecución del programa (interpretación).

Si hacemos click derecho sobre ParametrosDeEntrada y seleccionamos la opción void main(String arg[ ]) pasaremos a ejecutar el programa. En la ventana emergente pondremos lo siguiente: { "Pepe", "Luis", "Juan"} incluidos las llaves y las comillas dobles, los parámetros de entrada son Pepe, Luis y Juan. Cuando ejecutamos el programa se nos mostrará que la cantidad de parámetros son tres, y además nos mostrará el primer y segundo elementos del array, es decir, Pepe y Luis.

Si al ejecutar el método main de ParamentrosDeEntrada le pasaramos solo el parámetro "Pepe", como solo hay un solo parámetro de entrada, el array arg tiene un solo elemento. En este caso el programa nos va a dar un error porque la única variable que existe es arg [0], y al tratar de llamar a arg[1] nos va a dar error.

Veamos ahora qué ocurre al ejecutar ParametrosDeEntrada con los parámetros "Hola" "Java":




Estos dos parámetros son las cadenas "Hola" y "Java", las cuales son capturadas en las variables del array de cadenas arg[0] y arg[1]. Estas variables se usan para imprimir los parámetros en pantalla. Esto se puede ver en el código fuente del programa. Por otro lado, con el atributo “arg.length” podemos saber cuántos parámetros se le han pasado a nuestro programa. También decir que hacemos uso de un nuevo símbolo: “+”, que significa “concatenar” la cadena “Cantidad de parámetros” con el valor del atributo arg.length (que es 2 en este caso). El resultado es la cadena “Cantidad de parámetros : 2 “, que se imprime en la ventana como parte de la salida de nuestro programa.

El introducir parámetros al mismo tiempo que se invoca la ejecución de un programa es una posibilidad que admite Java. En otros lenguajes no es posible pasar parámetros de entrada a un programa.

**EJERCICIO 1**

Crea un array numérico con 5 elementos. Los números de cada elemento deben ser valores pedidos por teclado al usuario. Muestra por consola el índice y el valor al que corresponde. Debes utiliza bucles tanto para pedir los valores de los elementos del array como para mostrar su contenido por pantalla.

Ejemplo: Introducimos los valores 2, 4, 5, 8 y 10 .Lo que se tiene que mostrar por consola sería:

En el indice 0 esta el valor 2
En el indice 1 esta el valor 4
En el indice 2 esta el valor 5
En el indice 3 esta el valor 8
En el indice 4 esta el valor 10

Para comprobar si es correcta tu solución puedes consultar en los foros aprenderaprogramar.com

**EJERCICIO 2**

Crea un programa en el que se pida por consola el nombre de 2 alumnos y la nota de cada uno de ellos como valor numérico. El resultado que debe mostrarse es el nombre de cada alumno, su nota y su calificacion como texto (Sobresaliente, Notable, Bien o Suspenso).

Para ello crea un array numérico y otro de String (ambos unidimensionales). En el array numérico se insertarán las calificaciones facilitadas por el usuario entre 0 y 10 (debemos controlar que inserte una nota valida), pudiendo ser decimal. En el array de Strings se insertarán los nombres de los alumnos.

Crea también un array de String donde insertaremos el resultado de la nota con palabras admitiéndose estos valores: Sobresaliente, Notable, Bien o Suspenso.

El programa debe comprobar la nota de cada alumno teniendo en cuenta esta equivalencia:

Si la nota está entre 0 y 4,99 será un Suspenso.

Si la nota está entre 5 y 6,99 será un Bien.

Si la nota está entre 7 y 8,99 será un Notable.

Si la nota está entre 9 y 10 será un Sobresaliente.

Muestra por pantalla, el alumno su nota y su resultado en palabras utilizando bucles. Crea los métodos que creas convenientes. Puedes comprobar si tu código es correcto consultando en los foros aprenderaprogramar.com.

**Próxima entrega: CU00904C**

**Acceso al curso completo en aprenderaprogramar.com** -- > Cursos, o en la dirección siguiente:
http://aprenderaprogramar.com/index.php?option=com_content&view=category&id=58&Itemid=180