ARRAYS UNIDIMENSIONALES

Un array es un objeto en el que se puede almacenar un conjunto de datos de un mismo tipo. Cada uno de los elementos del array tiene asignado un índice numérico según su posición, siendo 0 el índice del primero.

Declaración

Un array debe declararse utilizando la expresión:

tipo [] variable_array;

o

tipo variable_array[];

Como se puede apreciar, los corchetes pueden estar situados delante de la variable o detrás. Los siguientes son ejemplos de declaraciones de array:

int[]k;
String[]p;
char cads[];

Los arrays pueden declararse en los mismos lugares que las variables estándar: como **atributos** de una clase o **locales** en el interior de un método. Como ocurre con cualquier otro tipo de variable objeto, cuando un array se declara como atributo se inicializa implícitamente al valor *null.*

Dimensionado de un array

Para asignar un tamaño al array se utiliza la expresión:

variable_array = new tipo [tamaño];

También se puede asignar tamaño al array en la misma línea de declaración de la variable. Los siguientes son ejemplos de dimensionado de un array:

```
k=new int [5];
cads=new char[10];
String [] noms=new String[10];
```

Cuando un array se dimensiona, **todos sus elementos son inicializados explícitamente** al valor por defecto del tipo correspondiente, independientemente de que la variable que contiene al array sea atributo o local.

Existe una forma de declarar, dimensionar e inicializar un array en una misma sentencia. La siguiente instrucción crea un array de cuatro enteros y los inicializa a los valores indicados entre llaves:

int[] nums = {10, 20, 30, 40};

Acceso a los elementos de un array

El acceso a los elementos de un array se realiza utilizando la expresión:

variable _array [índice]

Donde ***índice*** representa la posición a la que se quiere tener acceso, y cuyo valor debe estar comprendido entre 0 y *tamaño-1*.

Todos los objetos array exponen un atributo público, llamado ***length,*** que permite conocer el tamaño al que ha sido dimensionado un array. Este atributo resulta especialmente útil en aquellos métodos que necesitan recorrer todos los elementos de un array, independientemente de su tamaño.

El siguiente bloque de código utiliza ***length*** para recorrer un array y rellenarlo con números enteros pares consecutivos, empezando por el 0.

```
int [] nums=new int [10] ;
for (int i=0 ; i<nums.length; i++)
{
    nums [i] =i*2 ;
}
```

El programa que se muestra a continuación calcula la suma de todos los números almacenados en un array de enteros, mostrando además el mayor y el menor de los números contenidos:

```
public class Principal
{
   public static void main (String [] args)
   {
      int mayor, menor , suma;
      int [] nums = {3,4,8,2};

      suma = 0 ;
      menor = nums [0];
      mayor = nums [0];

      //en las variables mayor y menor
      //se obtendrá el mayor y el menor de los
      //números buscados, respectivamente. Ambas se
      //inicializan con cualquiera de los valores
      //del array
      //recorre el array en busca de los extremos,
      //acumulando en suma cada uno de los números
      //contenidos en el array

      for (int i=0 ; i<nums.length; i++)
      {
           if (nums[i] >mayor)
            {
                 mayor =nums [i];
            }
           if (nums[i] <menor)
            {
                  menor =nums[i];
            }
            suma+=nums[i];
```

```
        }
        System.out.println ("El mayor es "+ mayor);
        System.out.println ("El menor es "+ menor);
        System.out.println ("La suma es "+ suma);
    }
}
```

Paso de un array como argumento de llamada a un método

Además de los tipos básicos los arrays también pueden utilizarse como argumentos de llamada a métodos.

Para declarar un método que reciba como parámetro un array se emplea la sintaxis:

```
tipo método (tipo variable_array[])
{
   //código del método
}
```

El siguiente método recibe como parámetro un array de enteros:

```
void metodoEjemplo (int  m[])
{
    . . . .
}
```

En cuanto a la llamada, se utilizará la expresión:

```
objeto.método (variable_array);
```

A modo de ejemplo ilustrativo, la siguiente clase representa una versión del programa anterior en el que se calculaban la suma, el mayor y el menor de los números almacenados en un array de números. En este caso, se desglosa el código de las operaciones sobre el array en tres métodos a los que se les pasa como parámetro el array de enteros:

```
public class Principal{
      public static void main (String [] args) {
      int mayor, menor, suma;
      int [ ] nums = {3,4,8,2};
      //Llamada a los métodos
      suma= sumar (nums);
      menor=calculoMenor (nums);
      mayor=calculoMayor (nums);
      System.out.println  ("El  mayor  es  "+  mayor);
      System.out.print In ("El menor es "+ menor);
      System.out.println ("La suma es "+ suma);
}
static int sumar (int números []){
      int s=0;
      for(int i=0 ; i<numeros.length; i++) {
            s+=numeros[i];
      }
      return s;
}
static int calculoMayor (int números []){
      int may=numeros[0];
```

```
        for(int i=0; i<numeros.length;i++){
             if (nums[i] >may) {
                 may=nums [i];
             }
        }
            return may;
}
static int calculoMenor (int números []){
        int men=numeros [0] ;
        for (int i=0 ; i<numeros.length; i++) {
             if (nums [i] <men) {
                 men=nums [i];
             }
        }
        return men;
}
```

Array como tipo de devolución de un método

También un array puede ser devuelto por un método tras la ejecución del mismo. El formato de un método de esas características será el siguiente:

```
tipo [] método (parámetros) {

        .....
    return variable _array;
}
```

El siguiente método devuelve un array con los cinco números enteros siguientes al número recibido como parámetro:

```
int [] getNumeros (int n) {
        int [] nums=new int [5];
        for (int i=0 ; i<nums.length;i++){
            nums[i] =n+i+1;
        }
        return nums;
}
```

Recorrido de arrays *con for-each*

A partir de la versión Java 5, el lenguaje Java incorpora una variante de la instrucción *for,* conocida como ***for-each***, que facilita el recorrido de arrays y colecciones para la recuperación de su contenido, eliminando la necesidad de utilizar una variable de control que sirva de índice para acceder a las distintas posiciones. Aunque se utiliza un nuevo nombre para identificar esta instrucción *(for-each),* la palabra reservada para su implementación sigue siendo *for.*

En el caso de un array, el formato de la nueva instrucción for-each sería:

```
        for(tipo variable:var_array){
              //instrucciones
        }
```

donde, *tipo variable* representa la declaración de una variable auxiliar del mismo tipo que el array, variable que irá tomando cada uno de los valores almacenados en éste con cada iteración *del for,* siendo *var_array* la variable que apunta al array.

Por ejemplo, supongamos que tenemos el siguiente array:

```
int [] nums = {4, 6, 30, 15};
```

Y queremos mostrar en pantalla cada uno de los números almacenados en el mismo. Utilizando la versión tradicional *de for,* la forma de hacerlo sería:

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

Utilizando la nueva instrucción ***for-each***, la misma operación se realizará de la siguiente forma:

```
for(int n:nums) {
      System.out.println(n);
 }
```

Obsérvese cómo, sin acceder de forma explícita a las posiciones del array, cada una de éstas es **copiada automáticamente a la variable auxiliar n al principio de cada iteración.**

Como ejemplo ilustrativo, a continuación se presenta una nueva versión del programa anterior que realiza la suma de los elementos de un array. En este caso se han utilizado *bucles for-each* sin índices en vez *de for* tradicionales:

```
public class Principal {
   public static void main (String [] args) {
      int mayor, menor, suma;
      int [ ] nums = {3,4,8,2};
      //Llamada a los métodos
      suma=sumar (nums);
      menor=calculoMenor(nums);
      mayor=calculoMayor(nums);
      System.out.println("El mayor es "+ mayor);
      System.out.println("El menor es "+ menor);
      System.out.println("La suma es "+ suma);
   }
   static int sumar (int números []){
      int s = 0;
      for(int n: números ){
            s+=n;
      }
      return s ;
   }
   static int calculoMayor (int números []){
      int may=numeros [0];
      for(int n:numeros){
            if (n>may) {
                may=n;
            }
      }
```

```
        return may;
    }
    static int calculoMenor (int números []){
        int men=numeros [0];
        for(int n: números ){
                if (n<men) {
                        men=n;
                }
        }
        return men;
    }
}
```