Arrays

Los arrays (ver tema 1) se utilizan para crear listas y tablas. Aunque son **objetos**, su comportamiento es algo diferente al de los demás objetos de Java.

La principal característica de los arrays es que su tamaño es fijo, no pudiendo ser cambiado una vez creado el programa.

Constructores de los arrays

Los arrays tienen tres constructores, pero se escriben de forma diferente[1] a los constructores de las clases habituales.

1) El primer constructor es el que hemos visto en el tema 1. Podemos crear un array escribiendo sus datos encerrados entre llaves. Esto nos sirve cuando conocemos todos los datos que vamos a guardar en el array.

   - Por ejemplo, para crear una lista de números:

```
1.  int[] lista={4,9,-2,6};
```

   - Por ejemplo, para crear una tabla de números:

```
1.  int[][] tabla={
2.     {5,2,6,1},
3.     {2,3},
4.     {6,8,20,65,22}
5.  };
```

2) El segundo constructor nos permite crear un array del tamaño que indiquemos, pero todas sus casillas se rellenarán con el valor que dice esta tabla:

Tipo de dato	Valor por defecto
Números (int, long, double, etc)	0
boolean	false
Objetos (String, Mario, LocalDate, etc)	null

Este constructor sirve cuando queremos hacer un array cuyo tamaño conocemos, y más tarde cambiaremos su contenido (normalmente con un bucle).

   - Algunos ejemplos para crear listas con el segundo constructor serían:

```
1.  int[] lista=new int[50];  // crea una lista con 50 ceros
2.  boolean[] lista2=new boolean[1000]; // crea una lista con 1000 boolean
3.  String[] lista3=new String[100]; // crea una lista con 100 null
```

[1] Esto se hizo en su momento para facilitar a los programadores de C y C++ pasar a Java.

- También es posible crear tablas:

```
1.  int[][] tabla=new int[5][10]; // crea una tabla de 5 filas y 10 columnas rellena con 0
2.  LocalDate[] tabla2=new LocalDate[2][3]; // crea una tabla de 2 filas, 3 columnas rellena
     con null
```

3) El tercer constructor es una versión diferente del primer constructor. Por tanto, también sirve para crear un array cuyos elementos conocemos de antemano.

- Por ejemplo, para crear una lista de números:

```
2.  int[] lista=new int[] {4,9,-2,6};
```

- Por ejemplo, para crear una tabla de números:

```
6.  int[][] tabla=new int[][] {
7.      {5,2,6,1},
8.      {2,3},
9.      {6,8,20,65,22}
10. };
```

¿Qué ventaja tiene este constructor sobre el primero? Con este constructor podemos poner en líneas diferentes la declaración de la variable y luego la creación del array:

```
1.  int[] lista=null;   // creo la variable "lista", pero la dejo vacía
2.  lista = new int[]{4,9,-2,6};  // asigno a la variable "lista" un array de números
```

Si intentamos hacer esto con el primer constructor, tendremos un error.

Métodos de los arrays

Los arrays no tienen métodos (salvo los heredados de Object), por lo que lo único que podemos hacer con ellos es:

1) Obtener el total de elementos del array mediante **length**

- En una lista, nos sale el total de datos de la lista

```
1.  int[] lista={9,3,5,2};
2.  System.out.println(lista.length); // imprime 4 en la pantalla
```

- En una tabla, nos sale el número de filas.

```
1.  int[][] tabla= new int[9][26]; // tabla 9x26 rellena de 0
2.  System.out.println(tabla.length); // muestra 9, que es el número de filas
```

- En una tabla debemos aplicar length sobre cada fila para obtener el número de columnas de esa fila

```
1.  int[][] tabla= new int[9][26]; // tabla de 9 filas y 26 columnas rellena con 0
2.  System.out.println(tabla[0].length); // muestra 26, que es el nº columnas de la fila 0
```

2) Acceder al dato que ocupa una posición del array:

- En una lista, escribimos su posición entre corchetes

```
3.  int[] lista={9,3,5,2};
4.  System.out.println(lista[2]); // imprime 5 en la pantalla
```

- En una tabla, escribimos su fila y columna entre corchetes

```
3.  int[][] tabla= new int[9][26]; // tabla 9x26 rellena de 0
4.  System.out.println(tabla.length); // muestra 9, que es el número de filas
```

- En una tabla debemos aplicar length sobre cada fila para obtener el número de columnas de esa fila

```
1.  int[][] tabla= {
2.    {9,2},
3.    {4,-6,3}
4.  };
5.  System.out.println(tabla[1][2]); // muestra 3
```

3) Cambiar un dato del array

- En una lista, escribimos su posición entre corchetes

```
5.  int[] lista={9,3,5,2};
6.  lista[2]=16; // cambia el 5 por 16
```

- En una tabla, escribimos su fila y columna entre corchetes

```
5.  int[][] tabla= new int[9][26]; // tabla 9x26 rellena de 0
6.  tabla[3][5]=4; // cambia el 0 de la fila 3, columna 5 por un 4
```

1) Arrays

- **Librería:** *Java Collection Framework*
- **Archivo:** *Incluido en el JDK*
- **Desarrollador:** *Sun Microsystems*
- **Paquete:** *java.util*

Esta clase proporciona utilidades que facilitan el trabajo con los arrays.

Collections
+ static List<T> asList(T… datos)

- **asList:** Devuelve una lista con todos los elementos que se pasan como parámetros. La lista obtenida no es un ArrayList, sino que es una lista de solo lectura en la que solo se pueden leer los elementos.