1. Introducción a los arrays.

El concepto de array se verá en detalle en temas posteriores, pero resulta necesario hacer un pequeño acercamiento para algunos de los ejercicios que se plantean en esta unidad.

La idea es en sí sencilla, si cuando se define una variable en un programa asociamos un valor a un nombre (`int iEdad = 10;`). Reserva de espacio de memoria RAM de tamaño entero, donde queda asignado el valor 10 y dentro del programa está accesible con el nombre iEdad.

Mediante un array es posible identificar un conjunto de variables u objetos del mismo tipo bajo un nombre común. Así si definimos un array como `int [ ] iEdades = new int [10];` Estaremos reservando espacio en memoria un espacio de (10 * tamaño entero), donde albergar 10 valores bajo un mismo nombre iEdades. Pero, ¿cómo se diferencia cada uno? Se utiliza un índice a continuación del nombre, encerrado entre corchetes.

Veamos la sintaxis y uso de los arrays:

- **Declaración del array**. La declaración de un array consiste en decir "esto es un array" y sigue la siguiente estructura: "`tipo[ ] nombre;`". *El tipo será un tipo de variable o una clase ya existente*.
- **Creación del array**. La creación de un array consiste en decir el *tamaño* que tendrá el array , es decir, el *número de elementos que contendrá*, y se pone de la siguiente forma: "`nombre=new tipo [ dimension ]`", donde dimensión es un número entero positivo que indicará el tamaño del array . Una vez creado el array este no podrá cambiar de tamaño.

Veamos un ejemplo de su uso:

```
int [] n; // Declaración del array.
n = new int[10]; //Creación del array, reservando espacio en memoria.
int[] m=new int[10]; // Declaración y creación en una única sentencia.
```

Una vez hecho esto, ya podemos almacenar valores en cada una de las posiciones del array, usando corchetes e indicando en su interior la posición en la que queremos leer o escribir, teniendo en cuenta que la primera posición es la cero y la última el tamaño del array menos uno. En el ejemplo anterior, la primera posición sería la 0 y la última sería la 9 al ser su tamaño 10.

La modificación de una posición del array se realiza con una simple asignación. Simplemente se especifica entre corchetes la posición a modificar después del nombre del array. Veámoslo con un simple ejemplo:

```
int[] Numeros=new int[3]; // Array de 3 números (posiciones del 0 al 2).
Numeros[0]=99; // Primera posición del array.
Numeros[1]=120; // Segunda posición del array.
Numeros[2]=33; // Tercera y última posición del array.
```

El **acceso a un valor ya existente** dentro de una posición del array se consigue de forma similar, simplemente poniendo el nombre del array y la posición a la cual se quiere acceder entre corchetes:

```
int suma=Numeros[0] + Numeros[1] + Numeros[2];
```

Los arrays, como objetos que son en Java, disponen de un conjunto de propiedades muy útiles. Una de ellas es que **length** nos permite saber su tamaño. El siguiente ejemplo mostrará por pantalla "Longitud del array: 20"

```
int [] iNumeros = new int[20];
System.out.println("Longitud del array: "+Numeros.length);
```

Los arrays son herramientas Java que permiten implementar algoritmos muy potentes, principalmente cuando son utilizados con **sentencias iterativas**.

Otro uso muy habitual es en el paso de parámetros a métodos/funciones mediante arrays. Pero es un uso que todavía no hemos visto en el curso y lo trataremos más adelante. Un ejemplo sería;

```
int sumaarray (int[] j) {
int suma=0;
for (int i=0; i<j.length;i++)
suma=suma+j[i];
}
```