ARRAYS UNIDIMENSIONALES

Un array (arreglo) es una colección de valores de un mismo tipo engrosados en la misma variable. De forma que se puede acceder a cada valor independientemente. Para **Java** además un array es un **objeto** que tiene propiedades que se pueden manipular.

En **Java** se hace:

1. **Declaración del array**: un array unidimensional se hace con esta sintaxis**.**

```
Tipo_datos  identificador_array[];
```

También se puede declarar de esta forma:

```
Tipo_datos[]  identificador_array;
```

Ejemplo:

```
double cuentas[];      //Declara un array que almacenará valores  double
```

Esta declaración indica que tipo de datos contendrá el array, pero no reserva espacio en la memoria al no saberse todavía el tamaño del mismo.

2. **Inicialización del array**. Eso lo realiza el operador **new**, que es el que realmente crea el array indicando un tamaño. Cuando se usa **new** es cuando se reserva el espacio necesario en memoria. Un array no inicializado es un array **null.**

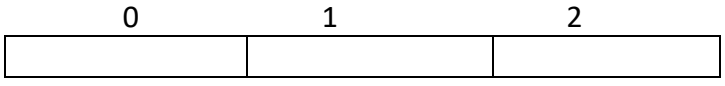
```
identificador_array[] = new tipo_datos[Numero_elementos];
```

Ejemplo:

```
int notas[];           //sería válido también int[] notas;
notas = new int[3];    //indica que el array tiene tres valores de tipo int
```

3. También se puede hacer todo a la vez, en una sola sentencia

```
int notas[]=new int[3];
```



En este ejemplo se crea un array de tres enteros (con los tipos básicos se crea en memoria el array y se inicializan los valores, los numéricos se inician a 0).

ASIGNACION DE VALORES AL ARRAY

1. Los valores del array se asignan utilizando el identificador del **array** seguido entre corchetes del índice.

```
notas[2]=8;
```

2. También se pueden asignar valores al array en la propia declaración:

```
int notas[ ] = {8, 7, 9};

int notas2[ ]= new int[ ] {8,7,9};      //Equivalente a la anterior
```

Se pueden declarar arrays a cualquier tipo de datos (enteros, booleanos, doubles, ... e incluso objetos).

Los arrays se numeran desde el elemento cero, que sería el primer elemento, hasta el tamaño-1 que sería el último elemento. Es decir, si tenemos un array de **diez elementos, el primer elemento sería el cero y el último elemento sería el nueve.**

3. Para asignar valores a todo el array se utiliza un bucle (for, while, do…while)

Ejemplo: para calcular la media de las notas de los 20 alumnos de un grupo:

```
int notas[] = new
int[20]; double suma=0;
for (int i=0;i<=19;i++){
      suma+=notas[i];                  //suma=suma+notas[i];
}
media=suma/20;
```

4. Un array se puede inicializar las veces que haga falta:

```
int notas[] = new notas[16];
notas = new notas[25];
```

Pero hay que tener en cuenta que el segundo **new** hace que se pierda el contenido anterior. Realmente un array es una referencia a valores que se almacenan en memoria mediante el operador **new**, si el operador **new** se utiliza en la misma referencia, el anterior contenido se queda sin referencia y, por lo tanto se pierde.

5. Un array se puede asignar a otro array (si son del mismo tipo), no copia los valores del array, son referencias al mismo array, y por lo tanto un cambio en cualquiera de los arrays provoca la modificación en el otro.

Ejemplo:

```
int notas[];
```

```
int ejemplo[]=new int[18];

notas=ejemplo;            //notas equivale a ejemplo.
```

Ejemplo:

```
int notas[]={3,3,3};
int ejemplo[]=notas; ejemplo=
notas; ejemplo[0]=8;
System.out.println(notas[0]);          //Escribirá el número 8
```

ARRAYS MULTIDIMENSIONALES

Los arrays además pueden tene varias dimensiones. Entonces se habla de arrays de arrays (arrays que contienen arrays)

1. Declaración de arrays bidimensionales:

```
int [ ][ ] a = { { 1 , 2 } , { 3 , 4 } , { 5 , 6 } }; //declaración e inicialización
```

Acceso a los elementos:

```
int x = a[1][0];          // contiene 3

int y = a[2][1];          // contiene 6
```

Otra forma de declarar los arrays es:

```
int [][]notas = new int[NF][NC];
```

Dónde:

- NF es el número de filas.
- NC es el número de columnas.

2. Otra forma de declarar los arrays es:

```
int notas[][];                  //notas es un array que contiene arrays de
enteros notas = new int[3][12];      //notas está compuesto por 3 arrays de 12
enteros cada uno notas[0][0]=9;             //el primer valor es 0
```

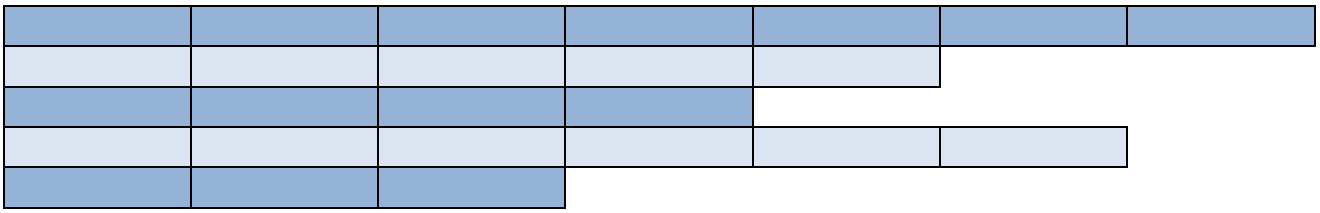
Puede haber más dimensiones incluso tres (notas[3][2][7]). Los arrays multidimensionales se pueden inicializar de forma más creativa incluso.

ARRAYS IRREGULARES

Son aquellos en los que cada fila puede tener distinto número de columnas.

Ejemplo:

```
int notas[][]=new int[5][ ];     // Hay 5 arrays de enteros

notas[0]=new int[7];             // El primer array es de 7 enteros

notas[1]=new int[5];             // El segundo de 5

notas[2]=new int[4];             // El tercero de 4

notas[3]=new int[6];            // El cuarto de 6

notas[4]=new int[3];             // El quinto de 3
```



Hay que tener en cuenta que en el ejemplo anterior, notas[0] es un array de 7 enteros. Mientras que notas es un array de 5 arrays de enteros.

Se pueden utilizar más de dos dimensiones si es necesario.

LONGITUD DE UN ARRAY

Los arrays poseen un método (función) que permite determinar cuantos elementos tiene un array: **length.**

Ejemplo :

```
int valores = new int[3];

System.out.println(valores.length);          //visualiza   3

int valores = new int[3][12];

System.out.println(valores.length);          //visualiza   3

System.out.println(valores[0].length);       //visualiza   12
```

ACCESO A LOS ELEMENTOS DEL ARRAY

Se pueden recorrer los elementos de un array multidimensional por filas, de la siguiente forma:

```
int [ ][ ] a = new int [3][2];
for (int f = 0 ;f < a.length ; f++)
   for (int c = 0 ; c < a[i].length ; c++)
            a[f][c] = f * c;
```

Obsérvese en el ejemplo la forma de acceder al tamaño de cada dimensión del array.

RECORRIDO DE LOS ELEMENTO POR COLUMNAS

```
for ( int c = 0 ; c < NC ; i++ )
     for ( int f = 0 ; f < NF ; f++)
            a[f][c] = f * c;
```

VISUALIZACION DEL ARRAY EN FORMA DE MATRIZ

```
for ( int f = 0 ;f < a.length ; f++ ){
     for ( int c = 0 ; c < a[i].length ; c++){
            System.out.print(a[f][c] +"\t");
     }
     System.out.println();
}
```