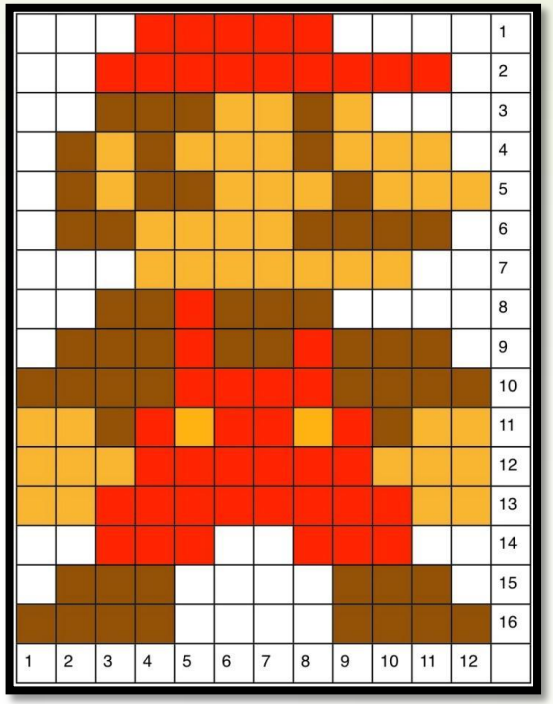
Desarrollo de Aplicaciones Web y Multiplataforma: **Programación**

DOCENTE: Daniel López Lozano

escuelaartegranada



Tema 3.

Estructuras complejas de datos: Array y Matriz

Índice de contenidos

- Tipo array ó vector.
- Array de arrays ó matriz.

- Los tipos simples o las librerías de Java no siempre son suficientes y es posible necesitar **tipos de datos personalizados** que se basen en los **ya existentes**.

- La **definición de tipos** de datos en la POO es **la base** de estos lenguajes.

- Un tipo de datos que es la **piedra angular** en un lenguaje de programación son los **Arrays** y son el primer **tipo de datos poderoso** que vamos a ver en este curso.

- Un Array es una **colección** de elementos de **un mismo tipo de datos** que nos permite manejar dichos elementos de forma conjunta.

- La colección de valores tienen alguna relación entre si.

- Supongamos que un programa necesita conocer las notas de **30 alumnos**. En lugar de crear 30 variables de tipo **double de forma individual**, se puede crear un **único Array** de tipo double cuyo tamaño sea igual a 30.

- En memoria RAM un Array es una **zona de almacenamiento continuo.**

- Lo habitual es que un Array tenga una **cantidad fija de memoria asignada**, que se establece al iniciar el programa.

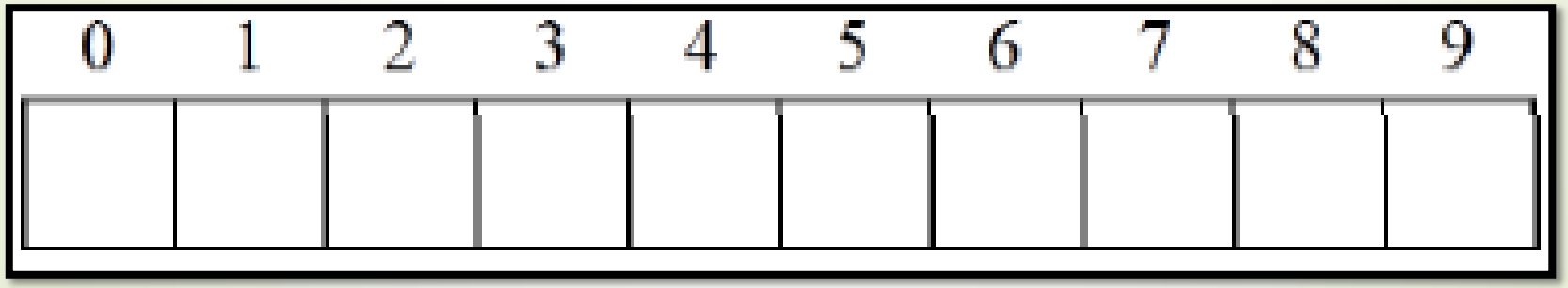
- Por ejemplo, si se crea una Array de tamaño 10, dicho Array **no podrá tener más** de 10 elementos.

- Cada elemento es referenciado **por la posición** que ocupa dentro del array.

- Dichas posiciones son **llamadas índices** y siempre son correlativos.

- El **primer elemento** del array será siempre **el 0** del mismo, es decir tendrá el índice 0.

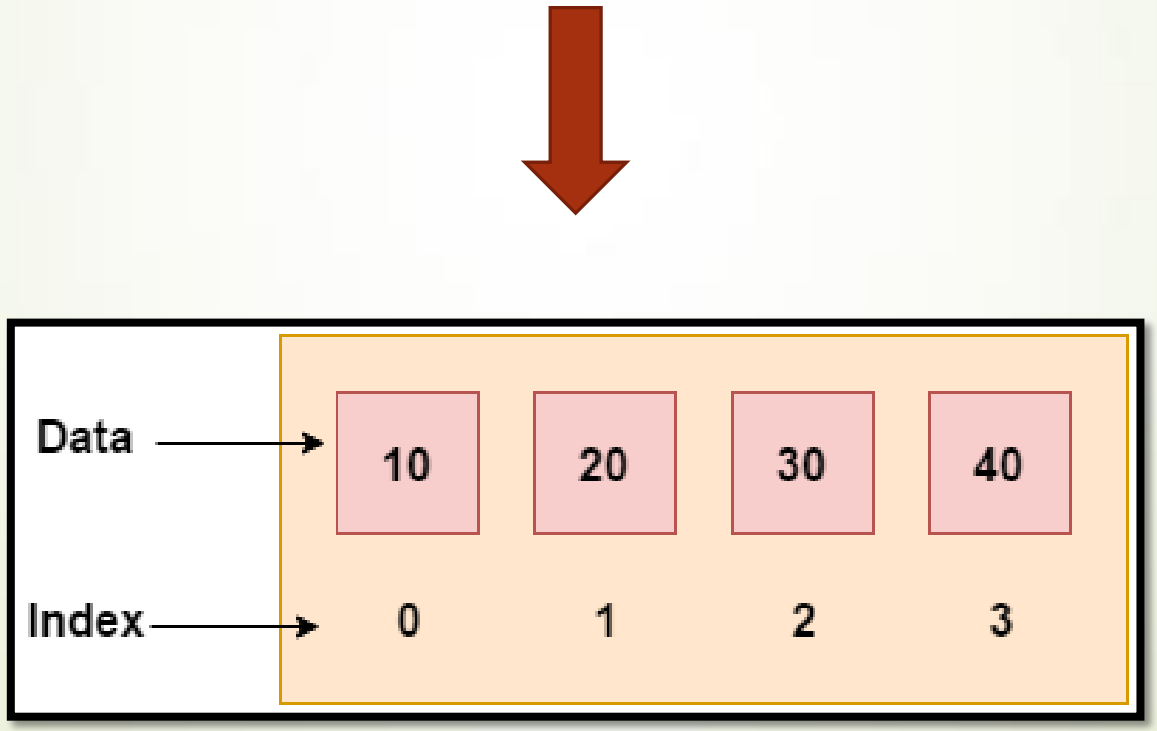
- En consecuencia si el array **tiene 'n' elementos**, el índice del último elemento ocupará la **posición 'n-1'**



- Para usar un Array hay que declararlo y usarlo dentro de **un bloque de código**.

```
//Declaracion de un array
 tipo_de_dato [] nombre_Array;
//Ejemplo
 int [] notasAlumnos;

//Creacion de un array
nombre_Array=new tipo_de_dato[tamaño];
//Ejemplo
notasAlumnos=new int[20];
//De una sola vez
int [] notasAlumnos=new int[20];
```

```
int [] numeros;
numeros=new int[4];
numeros[0]=10;
numeros[1]=20;
numeros[2]=30;
numeros[3]=40;
```



El porque de los Arrays resumido

```
//No es practico hacerlo asi
double alumno1;
double alumno2;
double alumno3;
double alumno4;
double alumno5;
...
double alumno20;

alumno1=sc.nextDouble();
alumno2=sc.nextDouble();
...

alumno20=sc.nextDouble();
```

```
//Mejor hacerlo así
double [] notas=new double[20];
for (int i=0;i<20;i++)
{
   notas[i]=sc.nextDouble();
}

//Usar mejor el notas.length en lugar de 20
```

Más ejemplos de Arrays

```
double [] x;
x = new double[100];

int [] v= new int[10];
int [] y= {0,1,2,3,4,5};

char [] w;
w = new char[4];
w[0]='h';
w[1]='o';
w[2]='l';
w[3]='a';
```

- Para usar un elemento determinado de un Array, simplemente hay que **indicar que posición** del Array se encuentra haciendo uso de **los []:**

```
System.out.println("La primera nota es "+notasAlumnos[0] );
System.out.println("La segunda nota es "+notasAlumnos[1] );
System.out.println("La tercera nota es "+notasAlumnos[2] );
System.out.println("La cuarta nota es "+notasAlumnos[3] );
...

int suma= notasAlumnos[0]+notasAlumnos[1]+notasAlumnos[2]+notasAlumnos[3]...;
System.out.println("La suma de las notas es"+ suma);
//¿Es la forma más acertada de hacerlo?
```

- Al crear un nuevo Array su tamaño no tiene que ser un número exacto si no que **puede ser una variable**.

- Además no tenemos que recordar que tamaño tenia cuando lo creamos, ya que el propio Array tiene un atributo **llamado length** que lo recuerda por nosotros.

```
int cantidad;
int [] numeros;

System.out.println("¿Cuantos numeros quieres?");
cantidad=sc.nextInt();
numeros=new int[cantidad];
```

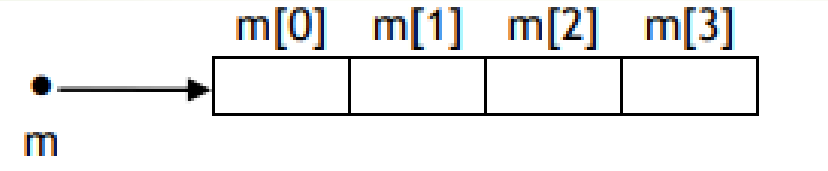
Completando el ejemplos de las notas

```
int alumnos;
double [] notasAlumnos;
double  media;

System.out.println("¿Cuantos alumnos tiene tu clase?");
alumnos=sc.nextInt();
notasAlumnos=new double[alumnos];
for (int i=0;i<notasAlumnos.length;i++)
{
  notasAlumnos[i]=sc.nextDouble();
}
for(int i=0; i<notas.length; i++){
    suma=suma+notasAlumnos[i];
}
media=suma/notasAlumnos.length;
System.out.print("La nota media de la clase es: "+media);
```

- Una aplicación interesante sobre todo para hacer pruebas con Arrays de gran tamaño **es llenarlo de números aleatorios**.

```
Random r=new Random();
int [] lista=new int[posiciones];

for(int i=0; i<lista.length; i++){
   lista[i]=r.nextInt(limite);
}
```

- Los array a la hora de almacenarse son referencias también denominados punteros en otros lenguajes.

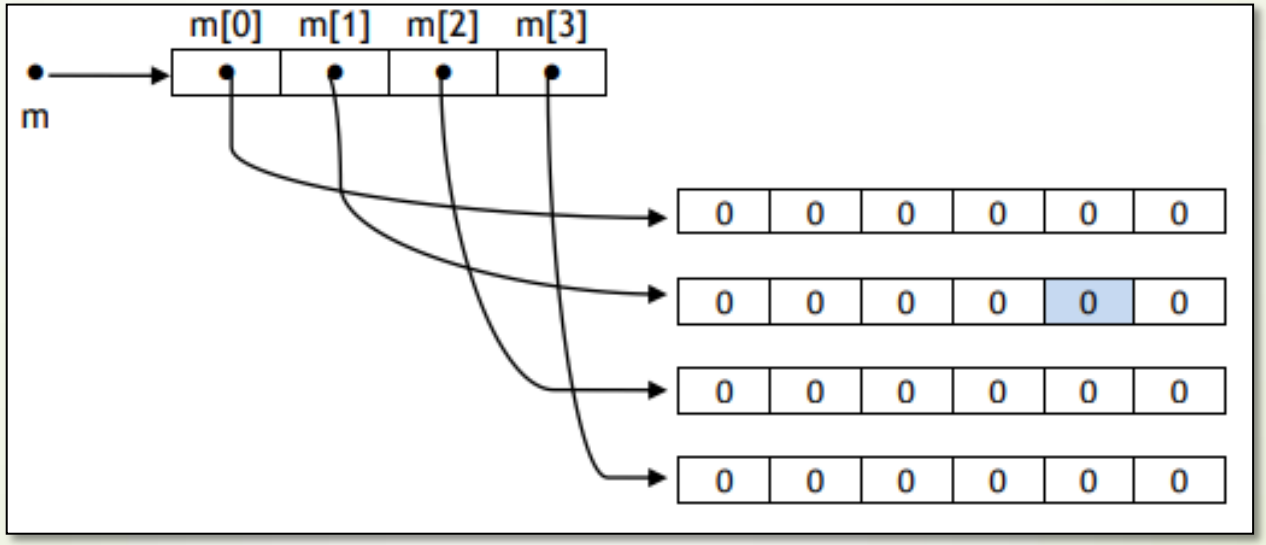


- Eso significa que son como objetos y por tanto no se pueden copiar con el = ni comparar con == o !=

- En el caso de los arrays tenemos que hacer dichas acciones a parte a mano.

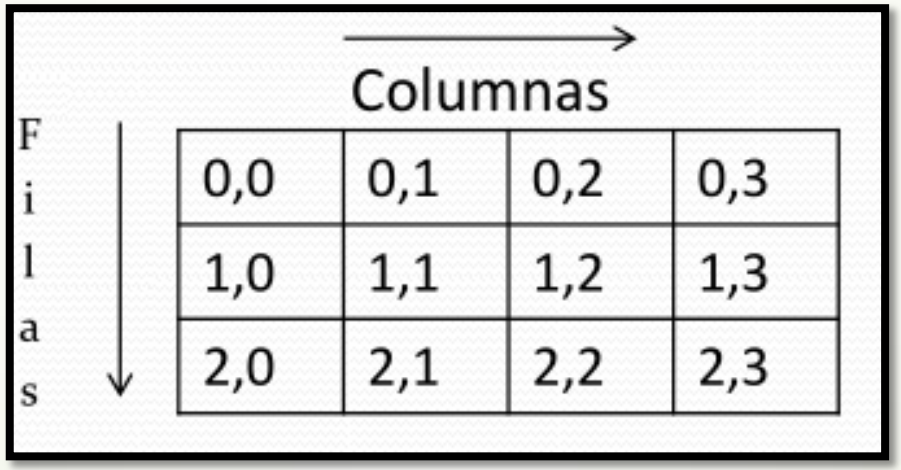
- En Java un **array** de 2 dimensiones (o matriz) es un array de arrays.

- Visto de manera interna se estructuran de la siguiente manera.



- Aunque internamente sea de una manera para nosotros van a funcionar como si fueran tablas de valores.

- Una matriz de enteros de 3x4 no es más que una matriz de 3 filas y 4 columnas.



- En una matriz, la primera coordenada es la fila y la segunda columna:

Nombre_matriz[filas][columnas]

- Podemos crear una matriz de las siguientes formas:

```
int [][] mat=new int[3][4];
int [][] otraMat={{0,0},{0,0},{0,0}}
```

- Para acceder a un elemento de la matriz hay que indicar sus dos coordenadas:

```
System.out.println("Elemento de la fila 1 columna 1: "+mat[0][0]);
System.out.println("Elemento de la fila 2 columna 3: "+mat[1][2]);
mat[0][2]=5;
```

- Para saber las dimensiones de un Array dado:

```
mat.length //Devuelve el número de filas de la matriz
mat[0].length: //Devuelve el número de columnas de la matriz
```

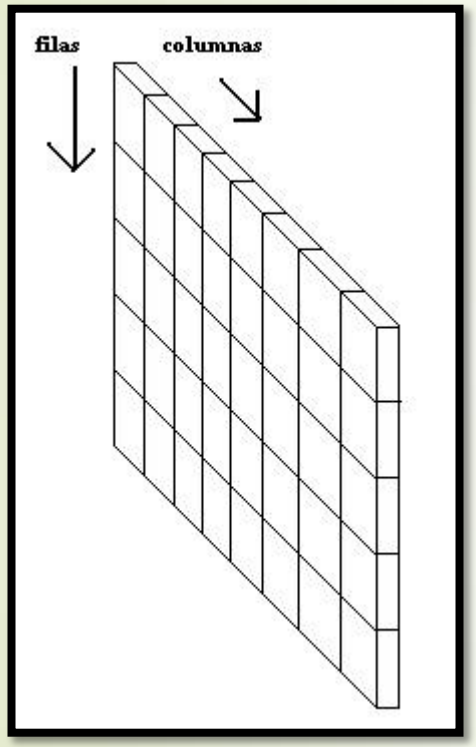
Códigos para manejar matrices

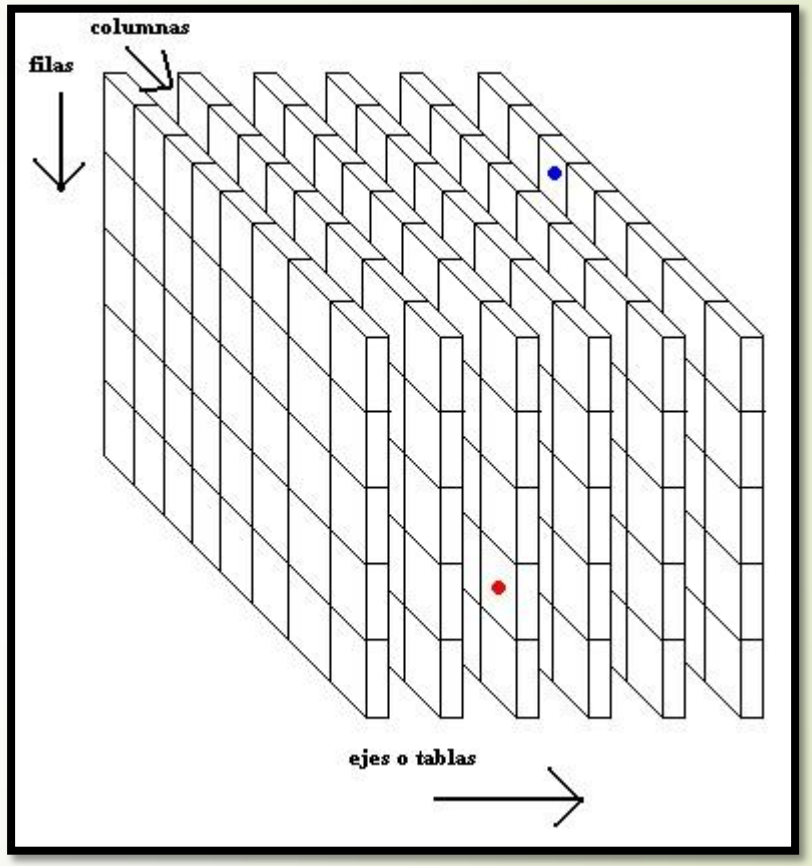
```
Random r=new Random();
int [][] mat=new int[filas][col];

for(int i=0; i<mat.length; i++){
    for(int j=0; j<mat[i].length; j++){
        mat[i][j]=r.nextInt(limite);
    }
}
```

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

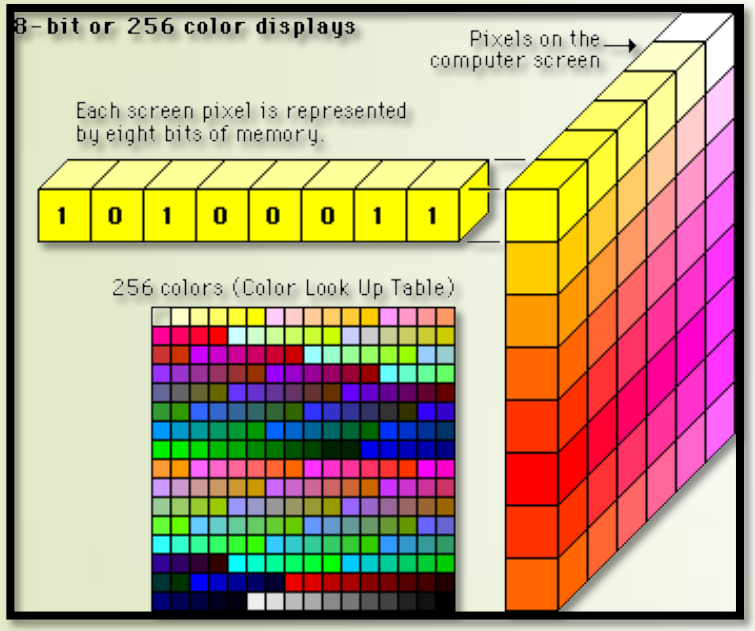
```
int [][] res=new int[s1.length][s1[0].length];

for(int i=0; i<res.length; i++){
    for(int j=0; j<res[f].length; j++){
        res[i][j]=s1[i][j]+s2[i][j];
    }
}
```

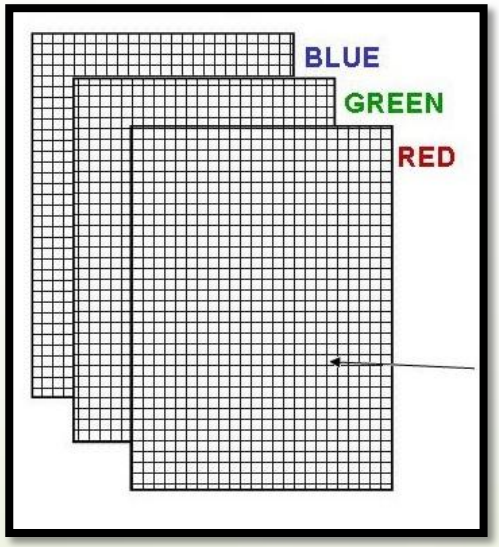
Matrices de más de 2 dimensiones

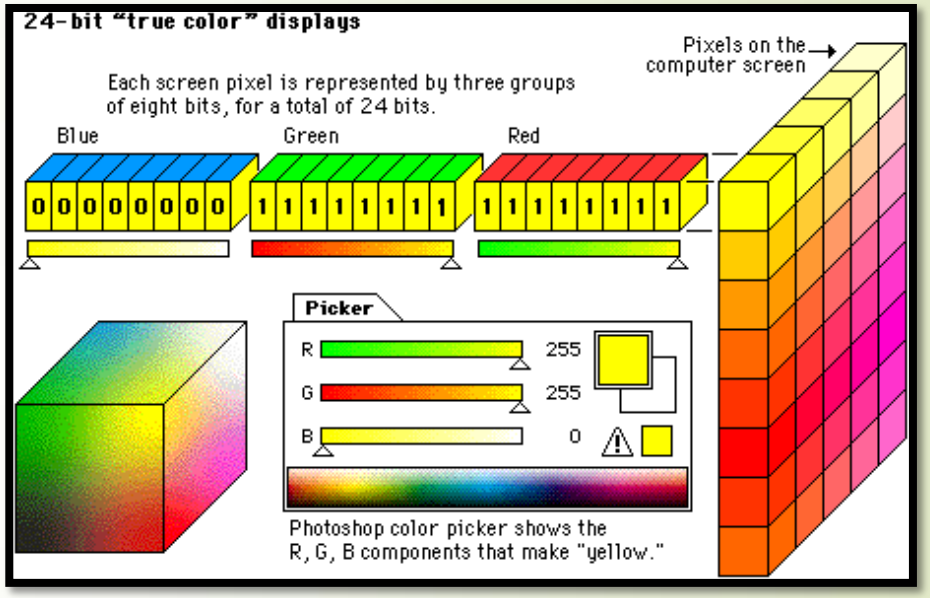




Matrices 3D entre nosotros







Bibliografía

- **García de Jalón, j.: "Aprende Java como si estuvieras en primero".** Editorial TECNUN. 2000
- **Holzner, S.: "La biblia de JAVA 2".** Editorial Anaya Multimedia 2000.
- **Moreno Pérez, J.C.: "C.F.G.S Entornos de desarrollo"** Editorial RA-MA. 2012
- **Wikipedia, la enciclopedia libre. http://es.wikipedia.org/** Última visita: Octubre 2018.
- **López, J.C.: "Curso de JAVA http://www.cursodejava.com.mx** Última visita: Octubre 2018**.**
- **Documentación oficial Java JSE 8 http://docs.oracle.com/javase/8/** Última visita: Octubre 2015.
- **Programación en castellano: Java. http://www.programacion.net/java Última visita: Octubre 2015.**