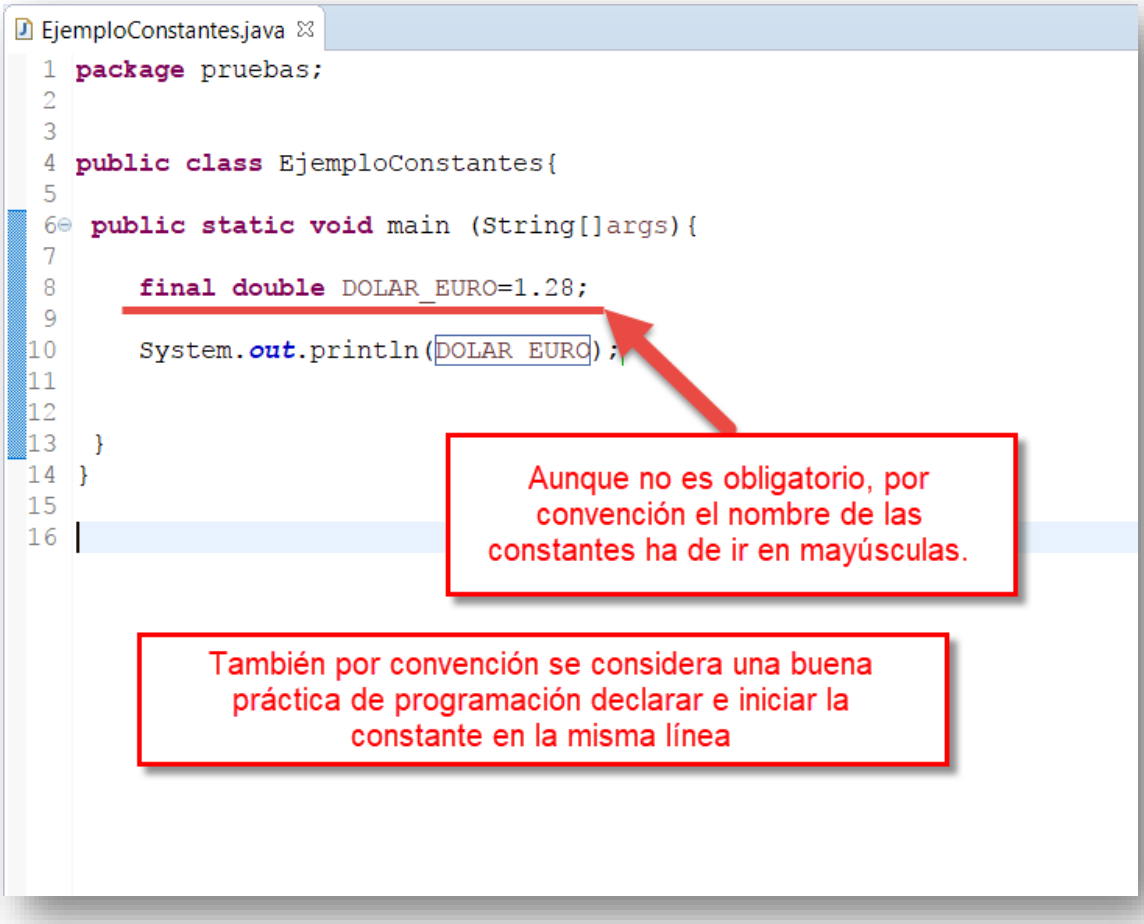
CONSTANTES Y OPERADORES

Qué son las constantes

La definición de **constante** es muy parecida a la definición de variable. Una **constante** es un espacio en la memoria del ordenador donde se almacenará un valor que no podrá cambiar durante la ejecución de un programa. Como se observa la definición es igual que la variable excepto por el matiz de que **el valor de una constante no puede cambiar durante la ejecución de un programa.**

En Java para denotar una constante se emplea la palabra reservada **final.** En la imagen que aparece a continuación se ve un ejemplo de declaración de constante:



La palabra reservada **final** indica que solo se puede dar valor a la variable una vez, y a partir de ese momento queda fijado de una vez por todas.

Operadores:

Los operadores habituales **+ - * /** se utilizan en Java para denotar suma, resta, multiplicación y división.

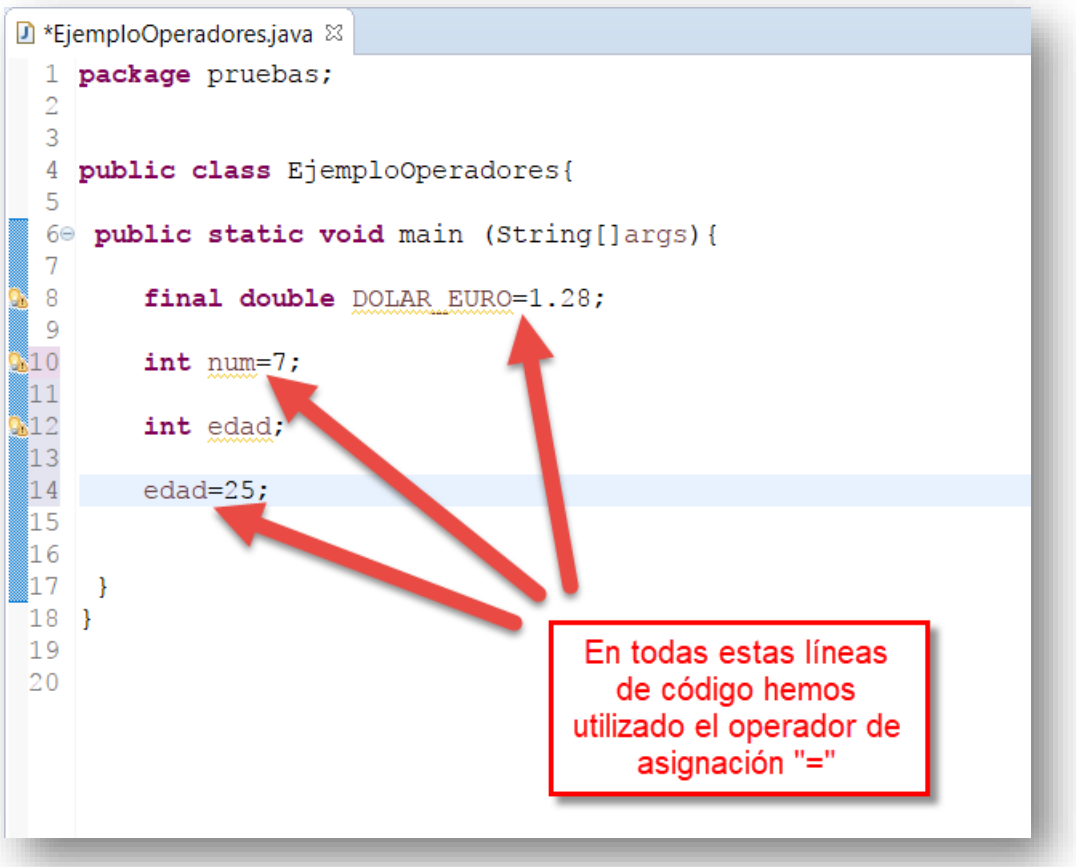
El resto entero, llamado **módulo**, se denota mediante el signo **%**. Por ejemplo **15 % 2 = 1.**

La división por cero da lugar a una excepción (un error), mientras que la división de un coma flotante por cero produce un resultado infinito o NaN (Not a Number).

Existe una abreviatura para emplear operadores en la asignación. Por ejemplo, x+=4; que es equivalente a x = x + 4;

- Operadores de asignación

El operador más importante y más frecuentemente usado es el operador asignación **=,** que hemos empleado para la inicialización de las variables y constanes. Así,



En la imagen anterior la primera sentencia declara una constante decimal de tipo **double** y le da un nombre (*DÓLAR_EURO*). La segunda sentencia usa el operador asignación para iniciar la

variable con el número 7. Y en la tercera sentencia, primero se declara la variable y después se inicia asignando el valor 25. Consideremos ahora, la siguiente sentencia

```
a=b;
```

que asigna a *a* el valor de *b*. A la izquierda siempre tendremos una variable tal como *a*, que recibe valores, a la derecha otra variable *b*, o expresión que tiene un valor. Por tanto, tienen sentido las expresiones

```
a=1234;

double area=calculaArea(radio);

superficie=ancho*alto;
```

Sin embargo, no tienen sentido las expresiones

```
1234=a;

calculaArea(radio)=area;
```

Las asignaciones múltiples son también posibles. Por ejemplo, es válida la sentencia

```
c=a=b;                    //equivalente a c=(a=b);
```

la cual puede ser empleada para inicializar en la misma línea varias variables

```
c=a=b=321;                //asigna 321 a a, b y c
```

- ## Operadores de incremento y decremento

Una de las operaciones más frecuentes en una variable numérica consiste en sumar o restar 1.

Java tiene operadores tanto de incremento como decremento para realizar estas sencillas operaciones.

n++ o n- - incrementa o decrementa en 1 el valor de la variable n.

Ejemplo: int n = 12; n ++;

En este ejemplo la variable n tiene un valor inicial de 12, pero inmediatamente después de le incrementa el valor en 1 pasando a tener un valor de 13.

Estos operadores no se pueden aplicar a valores numéricos. Por ejemplo 5++ no es una sentencia válida.

Hay dos formas de utilizar estos operadores, en forma de sufijo o en forma de prefijo. La única diferencia radica en utilizar el operador después del nombre de la variable (n++ forma sufijo) o

antes del nombre de la variable (++n forma prefijo). La única diferencia entre ambas formas surge cuando se utilizan los operadores dentro de una expresión.

Ejemplo: int m = 7; int n = 7; int a = 2 * ++m;

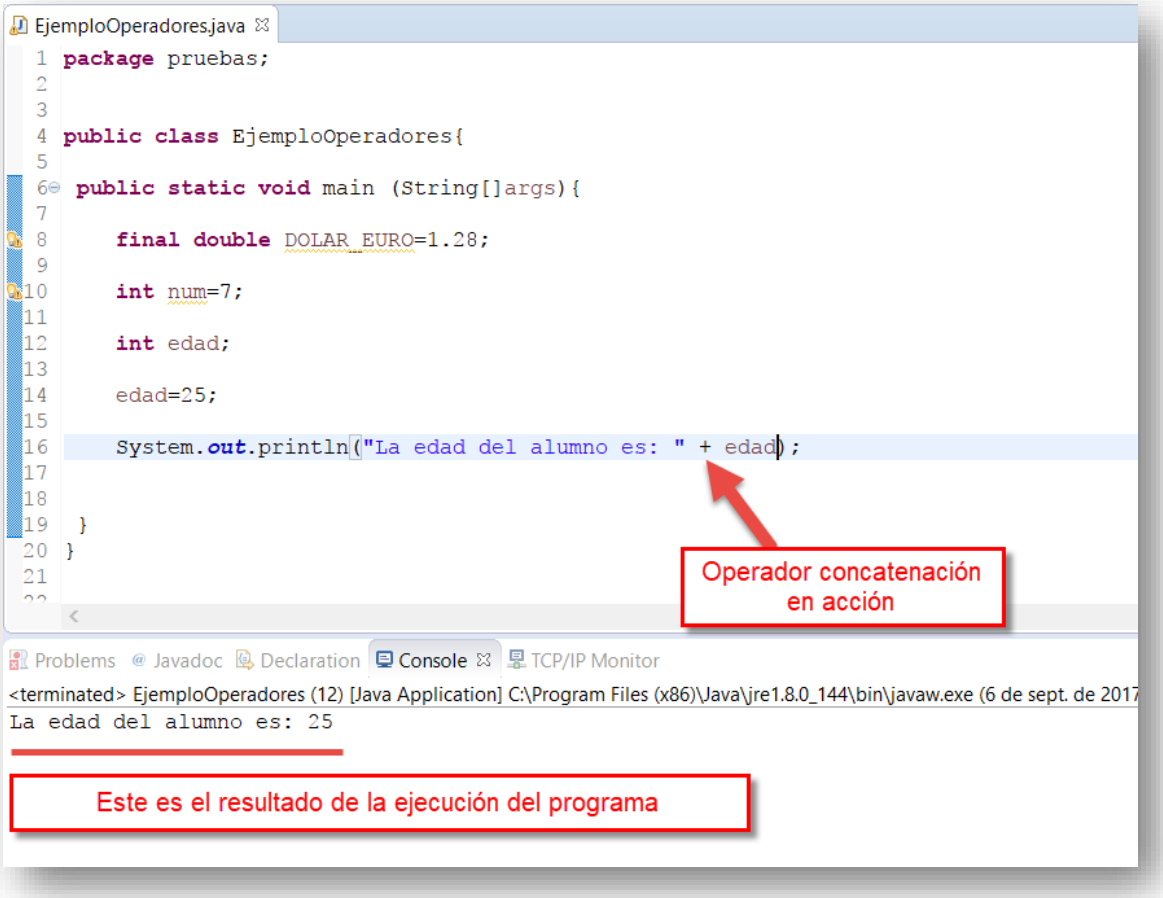
// en este caso a = 16 y m = 8

int b = 2 * n++

// en este caso b = 14 y n = 8

- Concatenación de Strings

En Java se usa el operador + para concatenar cadenas de caracteres o Strings. Ejemplo:



El operador + cuando se utiliza con Strings y otros objetos, creando un solo String que contiene la concatenación de todos sus operandos. Si alguno de los operandos no es una cadena, se convierte automáticamente en una cadena.

Como veremos más adelante, un objeto se convierte automáticamente en un string si su clase redefine la función miembro ***toString*** de la clase base ***Object*** (no te preocupes si no entiendes esto muy bien. Cuando llegues a la POO lo entenderás perfectamente).

- Operadores relacionales y booleanos

SIGNIFICADO	OPERADOR
Igual que	**==**
Distinto de	**!=**
Mayor que	**>**
Menor que	**<**
Mayor o igual que	**>=**
Menor o igual que	**<=**
Operador lógico Y	**&&**
Operador lógico O	**||**
Operador ternario	**Condición?expresión1:expresión2**