educación IT
CODE YOUR FUTURE

Introducción a Java

Módulo 1

Operadores

Operadores

Un **operador** es una expresión que produce **otro valor** así como las funciones o construcciones que **devuelven un valor.**

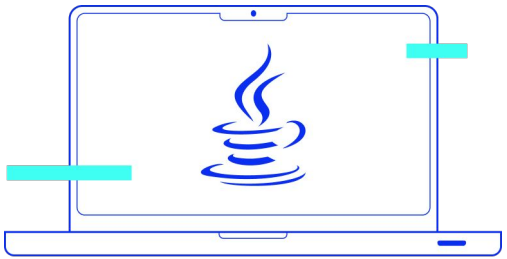
Existen operadores de *comparación*, *de negación*, *de incremento* y *decremento.*

Las operaciones matemáticas se comportan de igual manera en PHP. Las operaciones ***** y **/** tienen precedencia sobre la suma y la resta y se pueden utilizar **paréntesis** para expresiones más complejas.

Operadores en Java

Un **operador** es un símbolo especial que indica al compilador que debe efectuar una **operación matemática o lógica.**

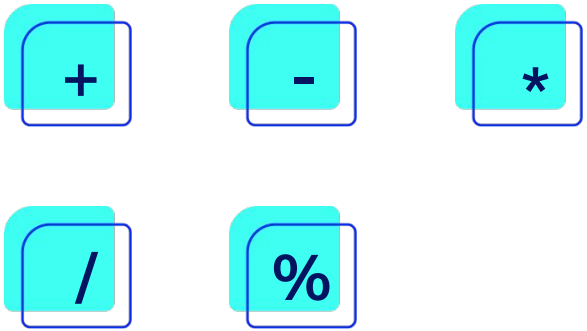
En todo lenguaje de programación existe un conjunto de operadores que permiten realizar operaciones con los datos. Nos referimos a operaciones aritméticas, comparaciones, operaciones lógicas y de otro tipo, por ejemplo, concatenación de cadenas, operaciones algebraicas entre valores, números, etc.



Tipos de operadores

- **Operadores aritméticos**

  Suma (+), resta (-), multiplicación (*), división (/), resto de la división (%).



- **Operadores lógicos**

  Los *operadores lógicos* o también llamados *de comparación* devuelven siempre ***verdadero (`true`) o falso (`false`)***. Este tipo de operadores se utilizan con mucha frecuencia en todos los lenguajes de programación para realizar una acción u otra en función de que cierta condición sea **verdadera o falsa**. Se utilizan para construir **expresiones lógicas combinando valores lógicos** *(`true` y/o).*

  En ciertos casos, el segundo operando no se evalúa porque ya no es necesario.

- **Operadores lógicos**

Operador	Utilización	Resultado
&&	op1**&&**op2	*true* si op1 y op2 son *true*. Si op1 es *false* ya no se evalúa op2.
||	op1**||**op2	*true* si op1 y op2 son *true*. Si op1 es *true* ya no se evalúa op2.
!	**!**op	*true* si op1 es *false* y *false* si op1 es *true*.
&	op1**&**op	*true* si op1 y op2 son *true*, pero siempre se evalúa op2.
|	op1**|**op2	*true* si op1 u op2 son *true*. Siempre se evalúa op2.

- **Operadores de asignación**

Operador	Utilización	Expresión equivalente
=	op1=op2	op1 = op2
+=	op1+=op2	op1 = op1 + op2
-=	op1-=op2	op1 = op1 - op2
*=	op1*=op	op1 = op1 * op2
/=	op1/=op2	op1 = op1 / op2
%=	op1%=op2	op1 = op1 % op2

- **Operador concatenación de caracteres**

  Se utiliza para concatenar cadenas de caracteres. Por ejemplo:

  ```
  "Se han comprado " + variableCantidad + "unidades";
  ```

- **Operador condicional ?:**

  También denominado ***inline-if.*** Uso:

  ```
  expresionBooleana ? res1 : res2
  ```

  Evalúa *expresionBooleana* y devuelve *`res1`* si el resultado es *`true`* o *`res2`* si el resultado es *`false`*. Ejemplo:

  ```
  a=100; b=50; String z = (a<b) ? “a es menor” : “a es mayor”;
  ```

- **Operadores incrementales y decrementales**
  - incremento (**++**)
  - decremento (**--**)

- **Operadores relacionales**

  Sirven para realizar **comparaciones de igualdad, desigualdad y relación de menor o mayor.** El resultado de estos operadores es siempre un valor booleano *(`true` o `false`).*

Operador	Uso	El resultado es *true*
>	op1>op2	Si op1 es mayor que op2
>=	op1>=op2	Si op1 es mayor o igual que op2
<	op1<op2	Si op1 es menor que op2
<=	op1<=op	Si op1 es menor o igual que op2
==	op1==op2	Si op1 y op2 son iguales
!=	op1!=op2	Si op1 y op2 son diferentes

- **Operadores aplicables a *bits***

Operador	Utilización	Resultado
>>	op1>>op2	Desplaza los bits de op1 a la derecha una distancia op2
<<	op1<<op2	Desplaza los bits de op1 a la izquierda una distancia op2
&	op1&op2	Operador AND a nivel de bits.
|	op1|op	Operador OR a nivel de bits.
^	op1^op2	Operador XOR a nivel de bits (1 si solo uno de los operandos es 1)
~	op1~op2	Operador complemento (invierte el valor de cada bit)

Clasificación

- **Operadores unarios:** aquellos que necesitan **un único operando**. Por ejemplo, el operador incremento (**++**) o el operador negación (**!**).
- **Operadores binarios:** precisan **dos operandos**. Por ejemplo, el operador suma (**+**) o el operador AND (**&&**).
- **Operadores ternarios:** demandan **tres operandos**. En Java, el único operador ternario es el operador condicional (**?:**).

Sentencias

Una **sentencia** es **una orden que se le da al programa para realizar una tarea específica**, como mostrar un mensaje en la pantalla, declarar una variable (para reservar espacio en memoria), inicializarla, llamar a una función, etc. Las sentencias acaban con **“;”**.

El carácter **“;” separa una sentencia de la siguiente**. Normalmente, las sentencias se ponen unas debajo de otras, aunque sentencias cortas pueden colocarse en una misma línea.

Aquí algunos ejemplos de sentencias:

```
int x=10;
```

```
import java.util.*;
```

```
System.out.println("Hola Mundo");
```

En el lenguaje Java, los caracteres espacio en blanco se pueden emplear libremente. Es muy importante para la legibilidad de un programa la colocación de unas líneas debajo de otras empleando **tabuladores**.

El editor del IDE nos ayudará plenamente en esta tarea sin apenas percibirlo.

educación IT
CODE YOUR FUTURE

¡Sigamos trabajando!