



Recibe una cálida:

¡Bienvenida!

Te estábamos esperando 😊 

» El entorno Java para la programación

Plan formativo: Desarrollo de Aplicaciones Full Stack Java Trainee V2.0

HOJA DE RUTA

¿Cuáles **skill** conforman el programa?



REPASO CLASE ANTERIOR

En la clase anterior trabajamos :

✓ Operadores en Java

LEARNING PATHWAY

¿Sobre qué temas trabajaremos?

4.3

Start! 🏁

El Entorno Java para la programación

Expresiones, bloques y sentencias.

Expresiones, sentencias y bloques en Java

Corrigiendo

OBJETIVOS DE APRENDIZAJE

¿Qué aprenderemos?



Comprender el concepto de las expresiones



Reconocer el concepto de sentencias



Aprender las implementaciones de los bloques



› Expresiones en Java



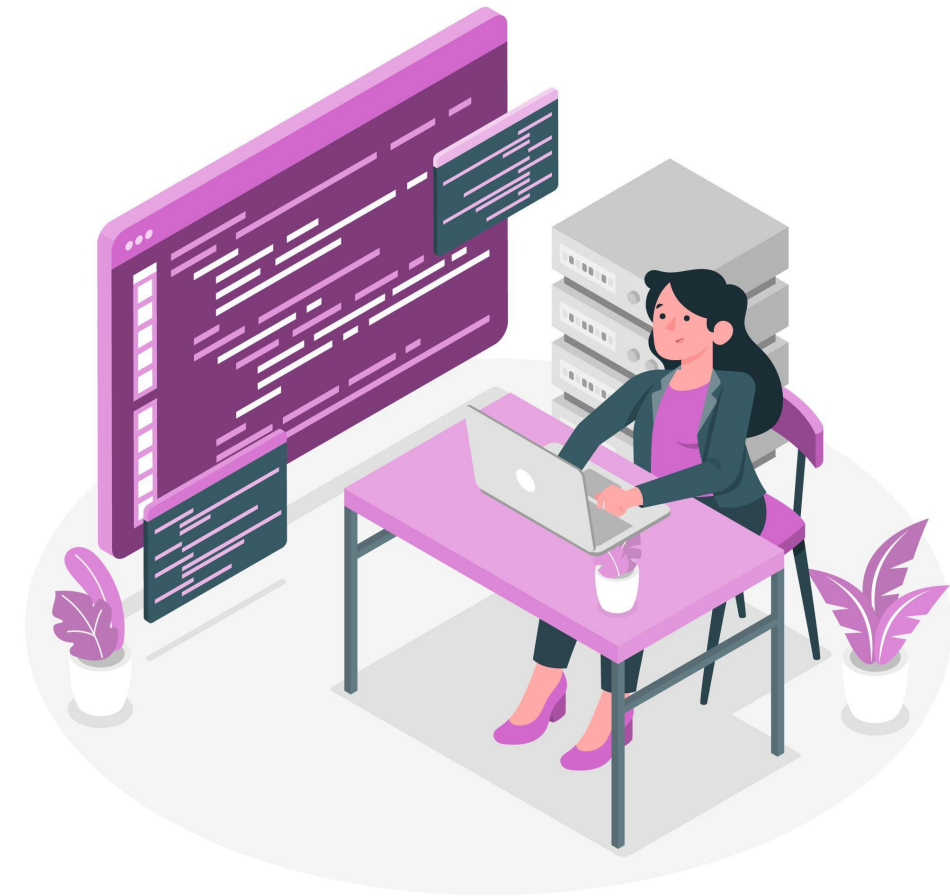
Expresiones



¿Qué es una expresión?

Una expresión es un conjunto de variables, operadores e invocaciones de métodos que se construyen para poder ser evaluadas retornando un resultado.

Cuando tengamos expresiones de evaluación complejas es recomendable que utilicemos paréntesis para saber cual es el orden de ejecución de operaciones. En el caso de no utilizar paréntesis se ejecutará el orden de preferencia de operadores. Por ejemplo, la división tiene más preferencia que la suma.





Expresiones

Si dos operadores se encuentran en la misma expresión, el orden en el que se evalúan puede determinar el valor de la expresión. En la siguiente tabla se muestra el **orden o prioridad en el que se ejecutan los operadores** que se encuentren en la misma sentencia. Los operadores de la misma prioridad se evalúan de izquierda a derecha dentro de la expresión.



Prior.	Operador	Tipo de operador	Operación
1	++ -- +, - ~ i	Aritmético Aritmético Aritmético Integral Booleano	Incremento previo o posterior (unario) Incremento previo o posterior (unario) Suma unaria, Resta unaria Cambio de bits (unario) Negación (unario)
2	(tipo)	Cualquiera	
3	*, /, %	Aritmético	Multiplicación, división, resto
4	+, - +	Aritmético Cadena	Suma, resta Concatenación de cadenas
5	<< >> >>>	Integral Integral Integral	Desplazamiento de bits a izquierda Desplazamiento de bits a derecha con inclusión de signo Desplazamiento de bits a derecha con inclusión de cero
6	<, <= >, >= instanceof	Aritmético Aritmético Objeto, tipo	Menor que, Menor o igual que Mayor que, Mayor o igual que Comparación de tipos
7	== i= == i=	Primitivo Primitivo Objeto Objeto	Igual (valores idénticos) Desigual (valores diferentes) Igual (referencia al mismo objeto) Desigual (referencia a distintos objetos)
8	& &	Integral Booleano	Cambio de bits AND Producto booleano
9	^ ^	Integral Booleano	Cambio de bits XOR Suma exclusiva booleana
10	 	Integral Booleano	Cambio de bits OR Suma booleana
11	&&	Booleano	AND condicional
12		Booleano	OR condicional
13	? :	Booleano, cualquiera, cualquiera	Operador condicional (ternario)
14	= *=, /=, %= +=, -= <<=, >>= >>>= &=, ^=, =	Variable, cualquiera	Asignación Asignación con operación

› Sentencias y Bloques en Java



Sentencias



¿Qué es una sentencia?:

Una sentencia es la **unidad mínima de ejecución de un programa**. Un programa se compone de uno o varios conjuntos de sentencias que acaban resolviendo un problema. En Java, **al final de cada una de las sentencias encontraremos un punto y coma (;)**.



- Sentencias de **declaración**: `int valor = 2;`
- Sentencias de **asignación**: `valor = 2;`
- Sentencias de **incremento** o **decremento**: `valor++;`
- Invocaciones a **métodos**: `System.out.println("Hola Mundo");`
- Creaciones de **objetos**: `Circulo miCirculo = new Circulo();`
- Sentencias de **control de flujo**: `if (valor>1) { ... };`



Bloques



¿Qué es un bloque en Java?:

Cuando hablamos de bloque, en programación, nos referimos a un **trozo de código que está delimitado** de alguna forma. **En Java, son unas llaves { }**. Una de apertura al principio y otra de cierre al final. Así, indicamos donde empieza ese bloque de código y dónde termina.



Los bloques son conjuntos de sentencias delimitados.






Bloques

Analicemos el código:

Las clases pueden contener entre sus llaves, muchas cosas, entre ellas, métodos, que es lo que empieza en la línea 2. Este método llamado main, es el encargado de ejecutar las instrucciones que hay en las líneas 3, 4, y 5.

El método, al igual que pasa con la clase, se abre en la línea donde se declara (línea 2) con una llave de apertura y se cierra cuando acaban las instrucciones que lleva escritas con la llave de cierre (línea 6).

El método es un bloque de código que está dentro de la clase, por este motivo, está envuelto entre las primeras llaves.



```
1 public class Main {  
2     public static void main(String[] args) {  
3         int numero1 = 7;  
4         int resultado = 10 + numero1;  
5         System.out.print(resultado);  
6     }  
7 }
```

LIVE CODING

Ejemplo en vivo

Practicando:

*Vamos a practicar un código simple para poner a prueba la jerarquía de las operaciones. Levantando la mano, los invitamos a contestar **¿Cuál operación se ejecuta primero? ¿Pueden dar otro ejemplo?***

1. `int resultado = 2 + 4 * 5;`
2. `int resultado = 2 + (4 * 5);`
3. `int resultado = 5 * 4 / 3;`
4. `int resultado = 5 * (4 + 3);`

Tiempo: 15 minutos



Ejercicio N° 1

Corrigiendo



Corrigiendo



Contexto: 🙌

Vamos a ver algunos bloques y sentencias de código a los cuales les faltan algunos delimitadores y operadores. De esta manera, vamos a practicar lo aprendido en la lección.



Consigna: ✍️

Coloca los signos, símbolos y operadores que faltan en el código.

Tiempo 🕒: 20 minutos





Corrigiendo 1

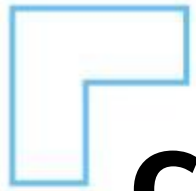


¡A corregir! 🙌 Completa el código: ¿Qué falta?



```
int edad = 15  
System.out.println(edad)
```





Corrigiendo 2

¡A corregir! 🙌 Completa el código: ¿Qué falta?



```
String nombre "Juan"
```

```
System.out.println nombre;
```





Corrigiendo 3

¡A corregir! 🙌 Completa el código: ¿Qué falta?



```
public static void main(String[] args) {  
    int edad 25;  
  
    System.out.println edad;  
}
```





Corrigiendo 4

¡A corregir! 🙌 Completa el código: ¿Qué falta?

```
1 public class Main
2     public static void main(String[] args)
3         byte numero1 = 10
4         byte numero2 = 20
5         boolean comparacion = numero1 != numero2
6         System.out.print(comparacion)
```





Corrigiendo

Ahora sí, a corregir: 🙌

Vamos a compartir las respuestas y también a abrir debate: **¿Cuál bloque fue más difícil de corregir? ¿Dónde tuvieron mayor facilidad? ¿Hubo alguno que no pudieron resolver?**

Respuestas: ✍️

- 1- Punto y coma en ambas líneas
- 2- Línea 1: operador =, punto y coma. Línea 2: () en nombre
- 3- Línea 1: operador = . Línea 2: paréntesis. Línea 3: cierre de llaves.
- 4- Línea 1: llaves de apertura. Línea 1: llaves de apertura. Línea 3, 4, 5 y 6: punto y coma. Línea 7: cierre de llaves.

Tiempo 🕒: 20 minutos



○

¿Alguna consulta?

+



RESUMEN

¿Qué logramos en esta clase?

- ✓ Reconocer la estructura de las expresiones.
- ✓ Comprender la estructura de las sentencias y bloques.



#WorkingTime

Continuemos ejercitando

¡Antes de cerrar la clase! Te invitamos a: 🙌 🙌 🙌

1. Repasar nuevamente la grabación de esta clase
2. Revisar el material compartido en la plataforma de Moodle (lo que se vio en clase y algún ejercicio adicional)
 - a. Material 1 (Foro)
 - a. *Lectura Módulo 4, Lección 3: páginas 5 - 6*
3. Traer al próximo encuentro, todas tus dudas y consultas para verlas antes de iniciar nuevo tema.

¡Muchas Gracias!

Nos vemos en la próxima clase 🙌



Momento: ✚

Time-out!

🕒 5 min.

