



CICLO 2

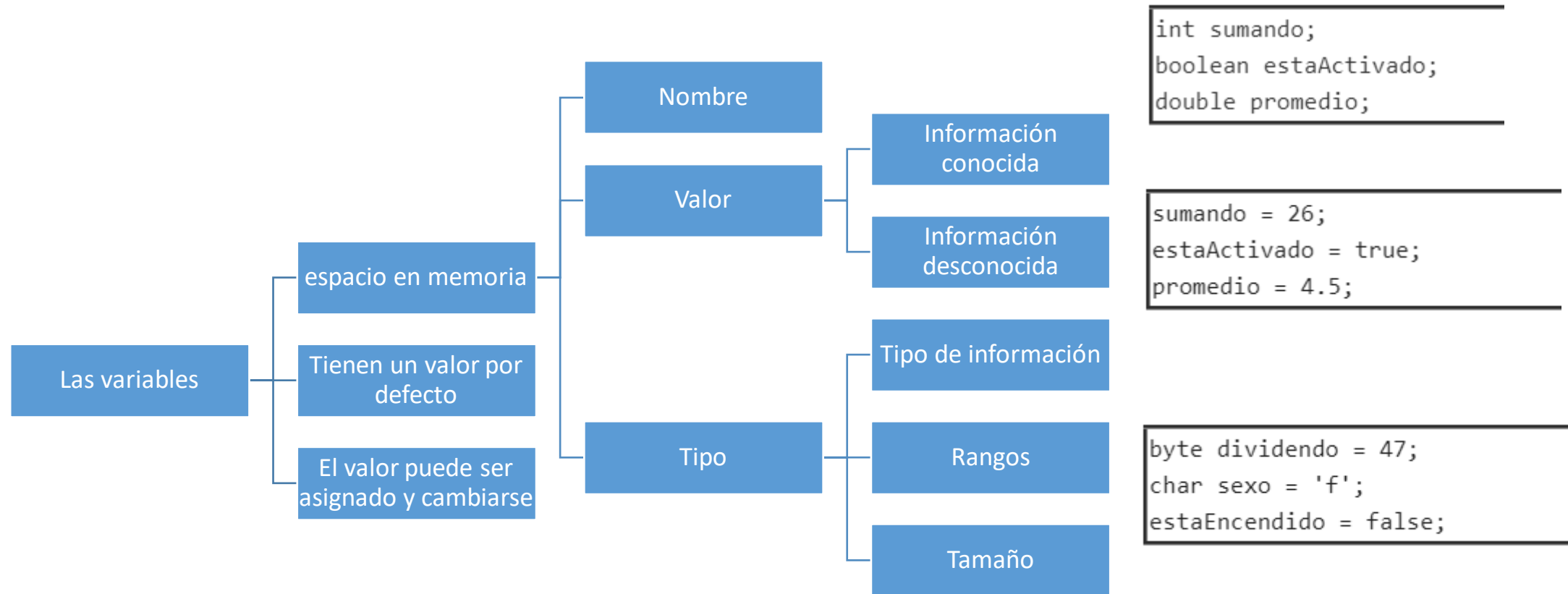
[FORMACIÓN POR CICLOS]

Programación Básica

Introducción al
lenguaje Java y su
Sintaxis



Las variables



Nombramiento de Variables

case-sensitive

- distinguen entre mayúsculas y minúsculas
- intereses**, **Intereses** e **INTERESES** son tres nombres de variable diferentes.

Puede comenzar por

- una letra
- el signo \$
- guion bajo (_)

No se permiten espacios en blanco.

Se recomiendan palabras auto-descriptivas

no sean una palabra reservada del lenguaje

se recomienda la notación *camel-case*

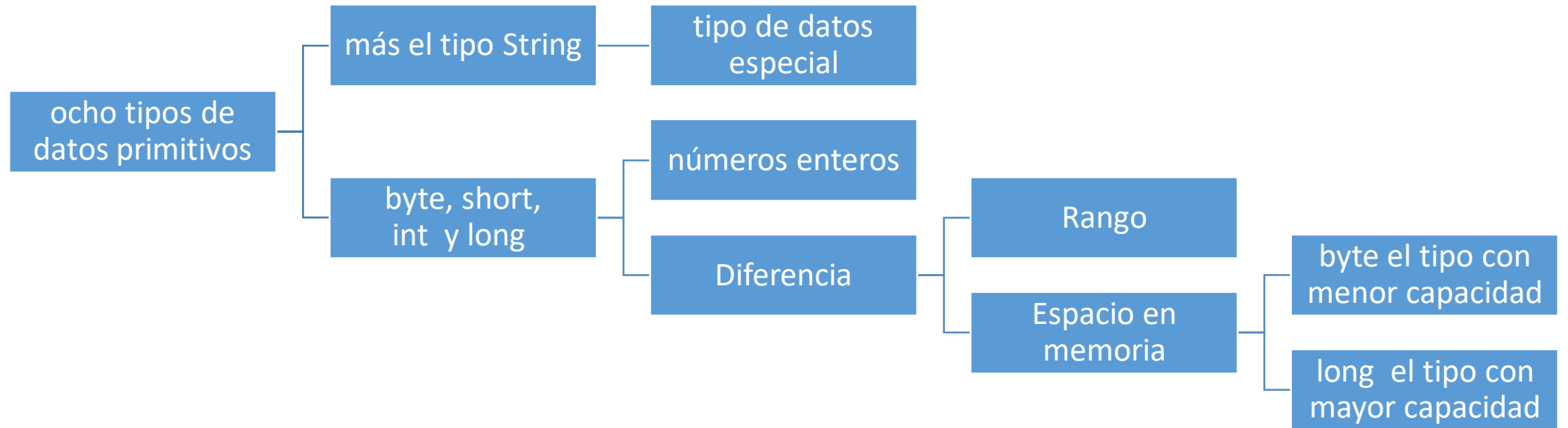
- una sola palabra
 - toda en minúsculas
- dos o más palabras
 - Primera letra en minúscula

```
int sumando;  
boolean estaActivado;  
double promedio;
```

```
sumando = 26;  
estaActivado = true;  
promedio = 4.5;
```

```
byte dividendo = 47;  
char sexo = 'f';  
estaEncendido = false;
```

Tipos de Datos Primitivos



Tipos de Datos Primitivos

float y double

- números de coma flotante (decimales)
 - double
 - ocupa más espacio que float
 - es un tipo más preciso

Boolean

- true (verdadero) o false (falso)

Char

- un solo carácter

String

- un arreglo o array de caracteres (char)

Tipos de Datos Primitivos

Tipo	Tamaño	Rango	Literal (ejemplo)
byte	8 bit	$[-128, 127]$	100
short	16 bit	$[-32.768, 32.767]$	10000
int	32 bit	$[-2^{31}, 2^{31}-1]$	100000
long	64 bit	$[-2^{63}, 2^{63}-1]$	1000001 ó 100000L
float	32 bit		4.25f ó 4.25F
double	64 bit		42.5 ó 42.5d ó 42.5D ó 4.25e1
boolean	1 bit	true / false	false
char	16 bit	$['\u0000', '\uffff']$	'a'

Operadores Básicos

determinan acciones que podemos hacer con variables de tipo primitivo

- aritméticos
 - realizar operaciones aritméticas con los números
 - concatenar cadenas de caracteres

los de comparación

- comparar valores y variables

lógicos

- permiten combinar comparaciones lógicas

Operadores	Precedencia
Postfix	expr++ expr--
Unario	++expr --expr +expr -expr ~ !
Multiplicativo	* / %
Aditivo	+ -
Shift	<< >> >>>
Relacional (lógico)	< > <= >= instanceof
Igualdad (lógico)	== !=
Bitwise AND	&
Bitwise exclusive OR	^
Bitwise inclusive OR	
AND lógico	&&
OR lógico	
Ternario	? :
Asignación	= += -= *= /= %= &= ^= = <<= >>= >>>=

Expresiones, Sentencias y Bloques

- **Expresiones**

- se compone de
 - Variables
 - literales (valores)
 - llamadas a métodos
 - operadores.

```
int frecuencia = 0;
unArray[0] = 100;
System.out.println("El primer elemento es: " + unArray[0]);
int resultado = (1 + 2) / 3;
if (valor1 == valor2) {
    System.out.println("valor1 es igual a valor2");
}
```

Expresiones, Sentencias y Bloques

- **Sentencia**

- Define una instrucción que damos usando el lenguaje y debe estar libre de ambigüedades.
- Terminan en ;

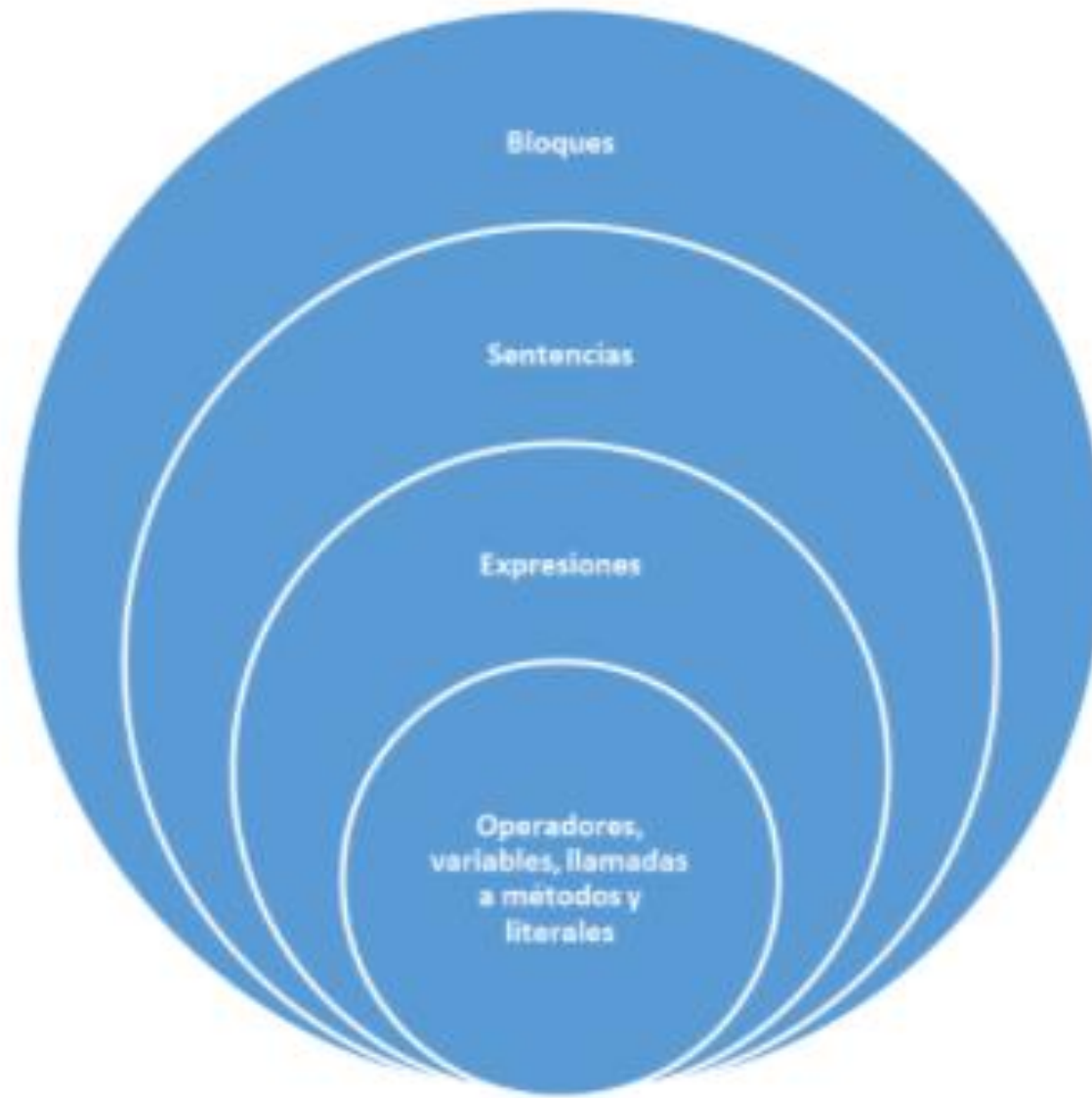
Tipos de sentencias	Ejemplos
Expresiones de asignación	<code>temperatura = 28;</code>
Uso de operadores de incremento	<code>temperatura++;</code>
Invocación de métodos	<code>System.out.println("Hace calor!!");</code>
Expresiones de creación de objetos	<code>Avion unAvion = new Avion();</code>
Sentencia de declaración	<code>float nota = 2.9;</code>
Sentencias de control de flujo	<i>Las veremos posteriormente</i>

Expresiones, Sentencias y Bloques

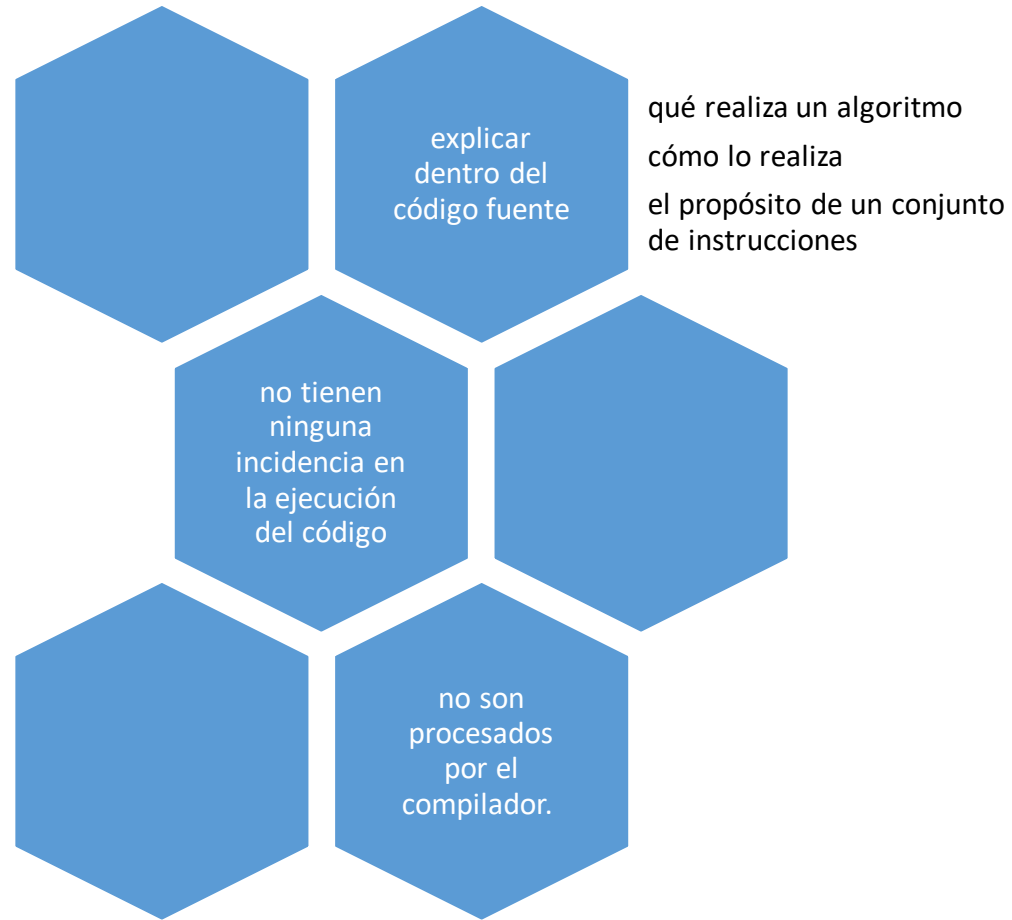
- **Bloques**

- conjunto de cero o más sentencias encerradas entre llaves ({})
- delimitar las sentencias
 - condicionales
 - de control de flujo (como los ciclos).

```
{  
    int c = 1.4 + 1.6;  
    System.out.println("La nota es: " + c);  
}
```



Comentarios



```
//Esto es un comentario de una sola línea
```

```
/* Esto es un comentario  
de más de  
una línea */
```

Cadenas de Caracteres String

cadena de caracteres

una secuencia de
caracteres

puede comprender

para las cadenas
de caracteres
usamos el
tipo **String**.

una palabra

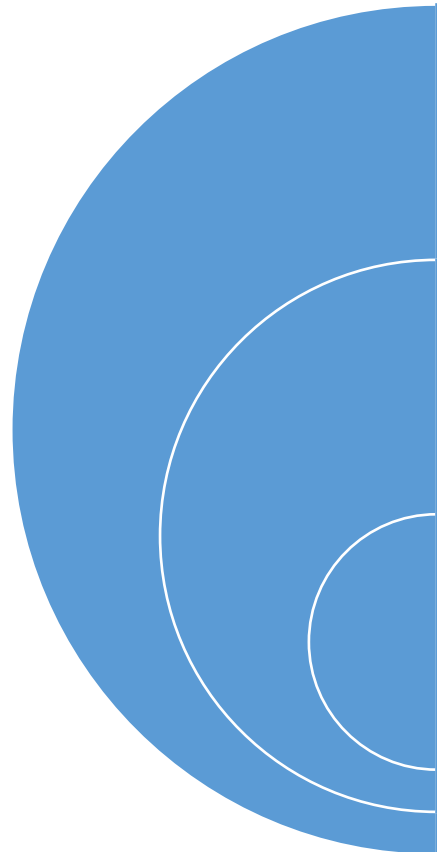
una frase

un párrafo

un libro

archivo con
código.

String



String	<ul style="list-style-type: none">• no es un tipo de datos primitivo• Es una clase
Permite usar una variable de ese tipo no solo como una variable	<ul style="list-style-type: none">• sino también como un objeto<ul style="list-style-type: none">• con sus métodos y propiedades.
Internamente, contiene un arreglo (o <i>array</i>) de caracteres	<ul style="list-style-type: none">• variables tipo char

Ejercicio*

Las sentencias if-then e if-then-else

```
if (condición) {  
    sentencia1;  
    sentencia2;  
    ...  
}
```


La sentencia if-then-else

```
if (condición) {  
    sentencia1;  
    sentencia2;  
} else {  
    sentencia3;  
    sentencia4;  
}
```