

VARIABLES Y CONSTANTES (Tipos Primitivos)

Las **variables** son contenedores de información que nos sirven para almacenar información en tiempo de ejecución que **puede cambiar**.

Las **constantes** realizan las mismas funciones que las variables, pero la información en tiempo de ejecución **no puede cambiar**. (SE HACEN IGUAL QUE LA TABLA DE VARIABLES DE ABAJO, PERO EN LA DECLARACIÓN SE PONE LA PALABRA RESERVADA "**final**" antes del tipo).

TIPO DE VARIABLE		DATO QUE ALMACENA	DECLARACIÓN (Definir tipo variable) (Solo una vez)	INICIALIZACIÓN (Dar valor a la variable)	VALOR POR DEFECTO (Sin inicializar la variable)
Boolean		True, False	boolean nombreVariable;	nombreVariable = true ;	false
ENTEROS	Byte	Entre -128 y 127	byte nombreVariable;	nombreVariable = 20 ;	0
	Short	Entre -32.789 y 32.767	short nombreVariable;	nombreVariable = 35000 ;	0
	Int	Entre -2 Millones y 2 Millones (Aprox)	int nombreVariable;	nombreVariable = 200000 ;	0
	Long	Entre -9x10 ¹⁸ y 9x10 ¹⁸	long nombreVariable;	nombreVariable = 400000000 ;	0
DECIMAL	Float	Entre -3,4x10 ³⁸ y 3,4x10 ³⁸	float nombreVariable;	nombreVariable = 4.57f ;	0.0f
	Double	Entre -1.79x10 ³⁰⁸ y 1.79x10 ³⁰⁸	double nombreVariable;	nombreVariable = 5.7894578d ;	0.0d
Char		Carácter en Unicode (Letra o símbolo del teclado)	char nombreVariable;	nombreVariable = ' H '; RECUERDA COMILLAS SIMPLES	'u0000'
String (Referenciado)		Cadena de caracteres (Varios char juntos)	String nombreVariable;	nombreVariable = "Hola" ; RECUERDA COMILLAS DOBLES	null

La declaración e inicialización se puede hacer en dos líneas como en la tabla superior o en la misma línea como en el ejemplo siguiente.

```
int nombreVariable = valor;
```

Se puede elegir cualquier nombre en una variable o constante, pero el lenguaje JAVA impone las siguientes normas

- El primer carácter tiene que ser una letra, "**_**" o "**\$**".

- No se permite espacios en el nombre
- No se permite el uso de palabras reservadas (palabras usadas en el propio lenguaje, en este documento están en azul claro).
 - No se permite símbolos extraños como por ejemplo “+” / “&”.
- El lenguaje JAVA distingue entre MAYUSCULAS y Minúsculas. (Palabra técnica para esto, **CASE SENSITIVE**).
 - Se permiten escribir caracteres del idioma español como la Ñ

Aunque JAVA no imponga muchas normas en el nombre el estándar de declaración de variables es el siguiente:

- Se empieza siempre por letra
- El nombre tiene que ser corto y debe tener significado con lo guarda.
 - Se puede usar números
- Si se quiere usar más de dos palabras, se colocan juntas y la primera letra en mayúscula menos la de la primera palabra. (Palabra técnica para esto, **DECLARACIÓN CAMEL**)

Ejemplo de variable declarada con este estándar

```
byte numeroLado1 = 12;
```

El estándar para la declaración de constantes es el siguiente:

- Se empieza siempre por letra
- El nombre tiene que ser corto y debe tener significado con lo guarda.
 - Se puede usar números
- Todas las palabras van en mayúsculas y se separan por un guion bajo.

Ejemplo de constante declarada con este estándar

```
final String DIRECCION_JUAN_1 = "Calle la amargura del programador"
```

Arrays (Tipo Referenciado)

En resumen, los Arrays son variables que pueden almacenar varios datos en tiempo de ejecución.

TIPO	DATO QUE ALMACENA	DECLARACIÓN (Definir tipo) (Solo una vez)	INICIALIZACIÓN (Dar valor a la variable)	VALOR POR DEFECTO (Sin inicializar la variable)
IGUAL QUE LAS VARIABLES	IGUAL QUE LAS VARIABLES	tipo nombreArray [] tipo nombreArray []=new tipo [5]	nombreArray []= {x, y, z, n} nombreArray[posición]=x;	IGUAL QUE LAS VARIABLES

ESTRUCTURAS DE CONTROL

CONDICIONALES

(Comprueban una condición lógica)

IF

```
if (condición) {  
    Instrucciones si se cumple la  
    condición;  
} else {  
    Instrucciones si no se cumple la  
    condición;  
}
```

ELSE IF (Condicionales Anidados)

```
if (condición 1) {  
    Instrucciones si se cumple la  
    condición 1;  
} else if (condición 2) {  
    Instrucciones si no se cumple la  
    condición 2;  
} else {  
    Instrucciones si no se cumple  
    ninguna condición;  
}
```

SWITCH CASE

```
switch (variable) {  
    case valor-n:  
        Sentencias;  
        break;  
    default:  
        Sentencias;  
}
```

BUCLES

(Repiten instrucciones x veces
(Interacción))

WHILE (Indeterminado)

```
while(condición) {  
    Instrucciones para repetir;  
    Incremento;  
}
```

DO WHILE (Indeterminado)

```
do {  
    Instrucciones para repetir;  
    Incremento;  
} while (condición)
```

FOR (Determinado)

```
for (variable, condición, incremento) {  
    Instrucciones para repetir;  
}
```

ESTRUCTURA DEL CODIGO Y OPERADORES

Estructura Básica

```
//Importación de clases
import java.*

//Declaración de clase principal
public class Nombre {

    public static void main (String []
    args) {

        //Método principal de acceso

        //Declaración de Variables

        Instrucciones y estructuras de
        control;

    }

}
```

Comentarios

Es una buena conducta del programador poner comentarios en el código para explicarlo y que el compilador/interprete no reconozca como código.

Para ello hay dos formas comentario en línea y comentario en párrafo.

```
//Hola soy un comentario en 1 línea
```

```
/*Hola soy un comentario que ocupa más
de una línea*/
```

OPERADORES

	Operador	Uso
Matemáticos	+	Sumar
	-	Restar
	*	Multiplicar
	/	Dividir
	%	Resto división
	++	Incremento +1
	--	Decremento -1

Relacionales	<	Menor que
	<=	Menor o igual que
	>	Mayor que
	>=	Menor o igual que
	==	Igual que
	!=	Distinto que

Lógicos	&&	AND (Si el primer dato es falso no evalúa el segundo)
	&	AND (Evalúa los dos datos)
		OR (No evalúa el segundo dato si el primero es verdadero)
		OR (Evalúa los dos)
	!	NOT