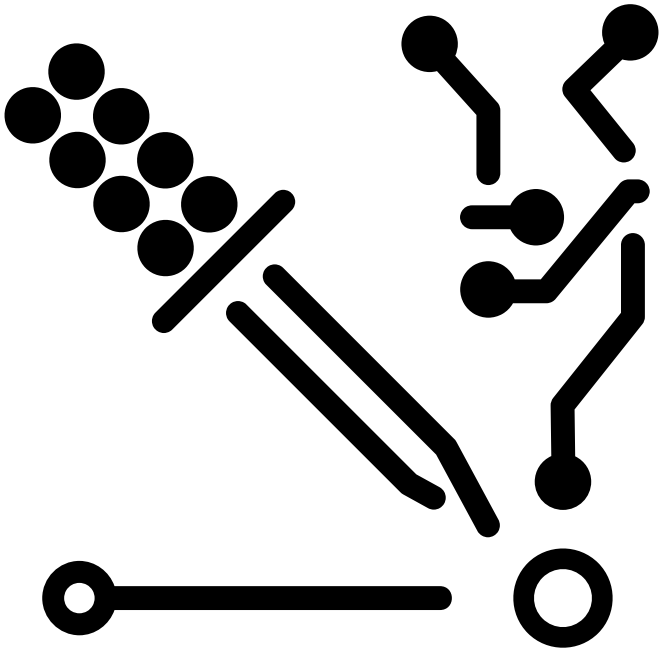


TALLER DE ROBÓTICA CON ARDUINO

CONSTANTES

Las constantes son expresiones predefinidas en el lenguaje Arduino que se utilizan para facilitar la lectura de los programas. Clasificamos las constantes en grupos:

- **Definición de niveles lógicos, constantes booleanas:**
  - true
  - false
- **Definición de niveles de pines:**
  - HIGH
  - LOW
- **Definición de modos de pines digitales:**
  - INPUT
  - INPUT_PULLUP
  - OUTPUT
- **Definición de incorporados:**
  - LED_BUILTIN

»»»» constants

CONSTANTES ENTERAS

Las constantes enteras son números que se usan directamente en un esquema, como 123. Normalmente, las constantes enteras se tratan como enteros en base 10 (decimales), pero se puede usar notación especial (formateadores) para ingresar números en otras bases.

BASE	EXAMPLE	FORMATTER	COMMENT
10 (decimal)	123	none	
2 (binary)	0b1111011	leading "0b"	characters 0&1 valid
8 (octal)	0173	leading "0"	characters 0-7 valid
16 (hexadecimal)	0x7B	leading "0x"	characters 0-9, A-F, a-f valid

»»»» Integer Constants

CONSTANTES FLOTANTES

Al igual que las constantes enteras, las constantes de punto flotante se utilizan para hacer que el código sea más legible. Las constantes de coma flotante se intercambian en tiempo de compilación por el valor al que se evalúa la expresión.

FLOATING-POINT CONSTANT	EVALUATES TO:	ALSO EVALUATES TO:
10.0	10	
2.34E5	2.34 * 10^5	234000
67e-12	67.0 * 10^-12	0.000000000067

»»»» Floating Point Constants

OPERADORES ARITMÉTICOS

=	Asignación
+	Adición
-	Sustracción
*	Multiplicación
/	División
%	Módulo

OPERADORES COMPUESTOS

++	Incremento
+=	Adición
--	Decremento
-=	Sustracción
*=	Multiplicación
/=	División
%=	Módulo

OPERADORES DE COMPARACIÓN

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

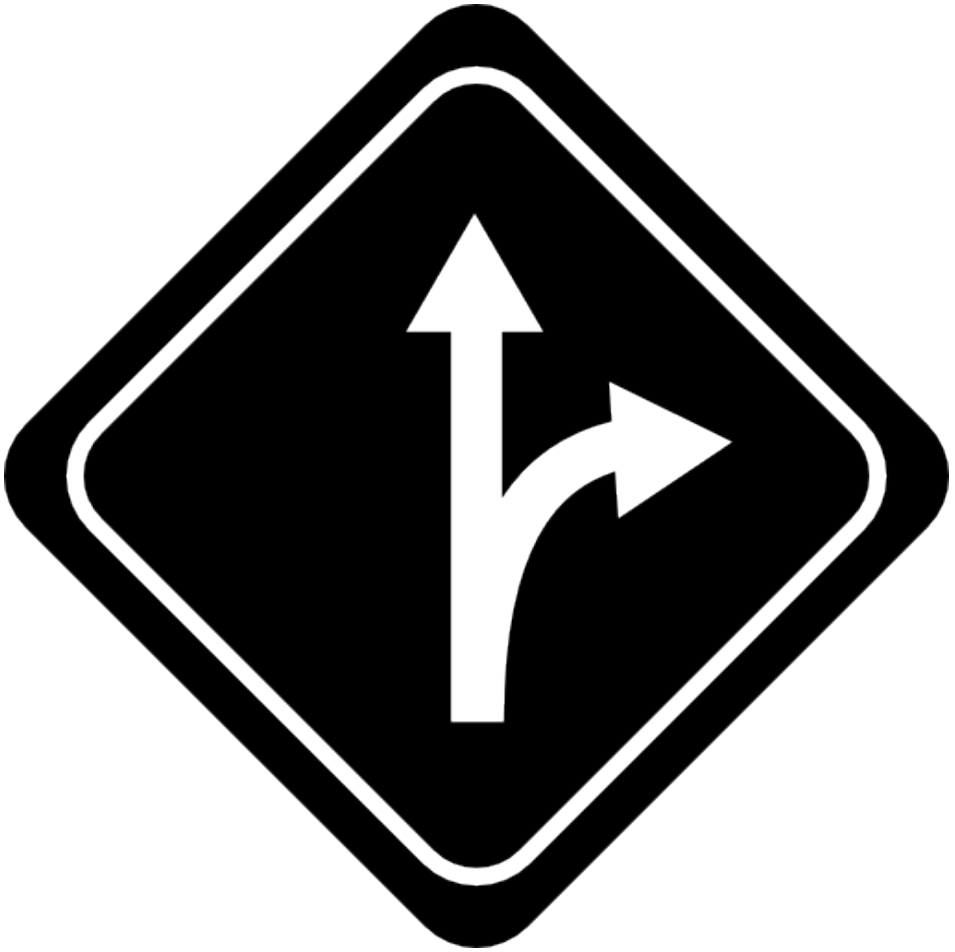
OPERADORES BOOLEANOS

!	NOT (NO)
&&	AND (Y)
||	OR (O)

PRECEDENCIA

Los operadores de relación tienen precedencia inferior que los operadores aritméticos, así que una expresión como **`i < lim-1`** se toma como **`i < (lim-1)`**, como se esperaría.
Las expresiones conectadas por **`&&`** o **`||`** son evaluadas de izquierda a derecha, y la evaluación se detiene tan pronto como se conoce el resultado verdadero o falso.

ESTRUCTURAS DE CONTROL



IF

```
if (condición) {
    // instrucción (es)
}
```

```
if (x > 120) {
    digitalWrite(12, HIGH);
    digitalWrite(13, HIGH);
}
```

X == Y (X es igual a Y)
X != Y (X no es igual a Y)
X < Y (X es menor que Y)
X > Y (X es mayor que Y)
X <= Y (X es menor o igual a Y)
X >= Y (X es mayor o igual a Y)

»»»» if

IF

Los corchetes pueden omitirse después de una declaración if. Si se hace esto, la siguiente línea (definida por el punto y coma) se convierte en la única declaración condicional.

```
if (x > 120) digitalWrite(LEDpin, HIGH);

if (x > 120)
digitalWrite(LEDpin, HIGH);

if (x > 120) {digitalWrite(LEDpin, HIGH);}

if (x > 120) {
  digitalWrite(LEDpin1, HIGH);
  digitalWrite(LEDpin2, HIGH);
}
```

IF

Y lógico:

```
if (condición 1 && condición 2) {
    // instrucción (es)
}
```

O lógico:

```
if (condición 1 || condición 2) {
    // instrucción (es)
}
```

IF ANIDADO

IF dentro de IF:

```
if (x > 120) {
    digitalWrite(12, HIGH);
    if (y > 25) {
        digitalWrite(13, HIGH);
    }
}
```

ELSE

```
if (condición) {
    // hacer A
}
else {
    // hacer B
}
```

```
if (x > 120) {
    digitalWrite(12, HIGH);
}
else {
    digitalWrite(12, LOW);
}
```

ELSE IF

```
if (condición 1) {
    // hacer A
}
else if (condición 2) {
    // hacer B
}
else {
    // hacer C
}
```

»»»» else

ELSE IF

```
if (temperatura >= 70) {
    digitalWrite(12, HIGH);
    digitalWrite(13, HIGH);
}
else if (temperatura >= 60) { // 60 <= temperatura < 70
    digitalWrite(12, LOW);
    digitalWrite(13, HIGH);
}
else { // temperatura < 60
    digitalWrite(12, LOW);
    digitalWrite(13, LOW);
}
```

SWITCH CASE

```
switch (variable) {
    case label1:
        // instrucciones
        break;
    case label2:
        // instrucciones
        break;
    default:
        // instrucciones
        break;
}
```

»»»» switch...case

SWITCH CASE

```
switch (var) {
    case 1:
        // hacer algo cuando var es igual a 1
        break;
    case 2:
        // hacer algo cuando var es igual a 2
        break;
    default:
        // si no coincide, hacer la opción por defecto
        // default es opcional
        break;
}
```

CRÉDITOS

Lucas Martín Treser
lmtreser@gmail.com – www.automatismos-mdq.com.ar