PROGRAMACION AVANZADA

ING. INDUSTRIAL

Profesora: Cristina Delia Cruz

FACULTAD DE INGENIERIA
UNIVERSIDAD NACIONAL DE JUJUY

ESTRUCTURA DE UN PROGRAMA EN C++

```
1 #include <iostream>
2
3 #include <stdlib>
4
5
6 [Declaración de Variables Globales]
7
8  int main()
9  {
10  [Declaración de Variables Locales]
11
12
13
14
15
16  }
```

Instrucciones Declarativas

Función Principal

Inicio del Programa

Cuerpo del Programa

Fin del Programa

ESTRUCTURA DE UN PROGRAMA EN C++

#include <Libreria _Solicitada>

Con esta directiva de preprocesador se incorporan las bibliotecas de funciones que se van a utilizar

Las librerías más utilizadas son:

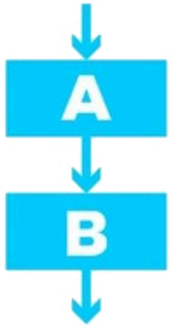
- **<iostream>** contiene las funciones para ingresar y mostrar datos.
- **<math.h>** contiene las funciones matemáticas comunes.
- **<time.h>** contienes las funciones para tratamiento y conversión entre formatos de fecha y hora.
- **<stdio.h>** contiene los prototipos de las funciones, macros y tipos para manipular datos de E/S.
- **<stdlib.h>** contiene tipos, macros y funciones para la conversión numérica, generación de
- memoria y tareas similares.
- **<string.h>** contiene los prototipos de las funciones y macros de clasificación de caracteres.

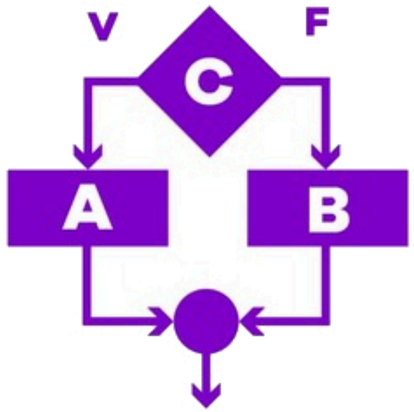
ESTRUCTURA DE UN PROGRAMA EN C++

Setencía

Definen la lógica de un programa o subprograma
Manipulan los datos para producir el resultado deseado por el usuario del programa
En programación estructurada, se tendrán sentencias que permiten:

- Estructura secuencial
- Estructura selectiva
- Estructura repetitiva





Dato

Un dato es cualquier valor o pieza de información que un programa puede utilizar.
Estos datos pueden ser números, texto, fechas, imágenes, etc.
Para que la computadora pueda trabajar con estos datos, es necesario definir su tipo y cómo se representarán en la memoria.
Los datos en un programa se pueden representar a través de:

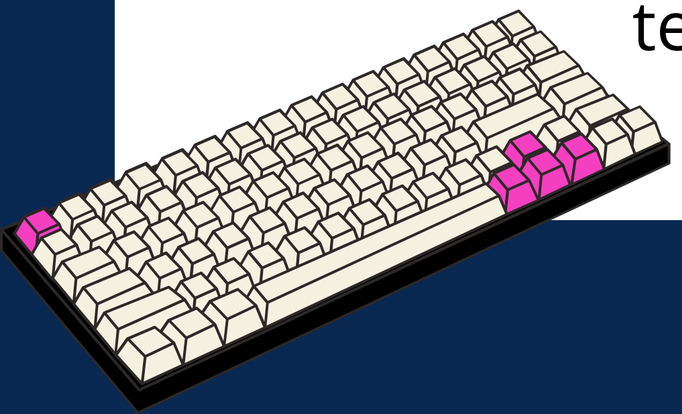
- Variables
- Constantes


```
1   #include <iostream>
2
3   using namespace std;
4
5   int main()
6   {
7       int a;
8       float b = 4.7;
9       char c = 'k';
10      bool d = true;
11      return 0;
12  }
```

TIPOS DE DATOS EN C++

DATOS SIMPLES EN C++

C++ es un lenguaje de programación que hereda muchos conceptos del lenguaje C, es un lenguaje compilado y fuertemente tipado, lo que significa que en las variables con las que trabajamos hay que indicar el **tipo del dato** que van a guardar cuando se declaran, lo que puede hacer que tengamos problemas y se generen errores.



TIPOS DE DATOS EN C++

Nombre	Descripción	Tamaño*	Rango de valores*
char	Carácter o entero pequeño	1byte	con signo: -128 to 127 sin signo: 0 a 255
short int (short)	Entero corto	2bytes	con signo: -32768 a 32767 sin signo: 0 a 65535
int	Entero	4bytes	con signo: -2147483648 a 2147483647 sin signo: 0 a 4294967295
long int (long)	Entero largo	8bytes	con signo: -2147483648 a 2147483647 sin signo: 0 a 4294967295
bool	Valor booleano. Puede tomar dos valores: verdadero o falso	1byte	true o false
float	Número de punto flotante	4bytes	3.4e +/- 38 (7 digitos)
double	De punto flotante de doble precisión	8bytes	1.7e +/- 308 (15 digitos)
long double	Long de punto flotante de doble precisión	8bytes	1.7e +/- 308 (15 digitos)

DECLARACION DE VARIABLE EN C++

Es preciso declarar las variables antes de utilizarlas.

La sintaxis de declaración es:

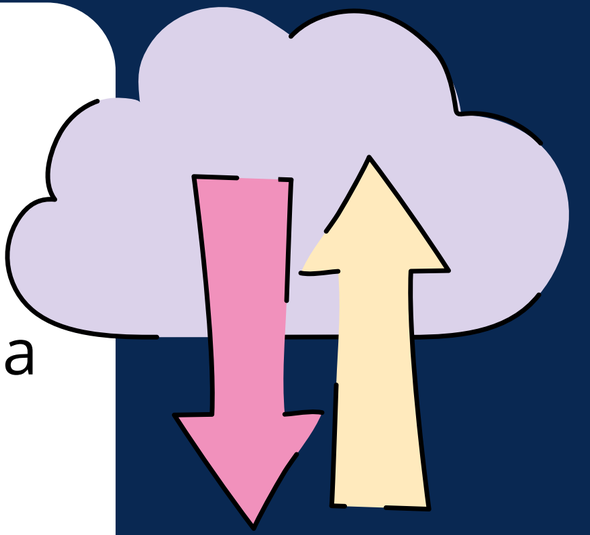
```
tipo nombre_de_variable[=valor];
```

Lo que está entre corchetes es opcional, se puede o no dar el valor a la variable al mismo tiempo que se la declara. Además, se puede declarar varias variables del mismo tipo separándolas con comas. Si se quisiera declarar variables como las básicas, sería así:

1. `int T=1401, variable=1, a;`
2. `float pi=3.14, decimal, otra=0.23;`
3. `char letra='C', mas;`
4. `bool dicho=true, bandera=false;`

Nota: después de declarar una serie de variables de un mismo tipo hay que poner un **;**

ENTRADA Y SALIDA EN C++



Las bibliotecas estándar de C++ proporcionan un amplio conjunto de capacidades de Entrada/Salida (E/S). C++ utiliza E/S a prueba de tipos. Cada operación de E/S se realiza automáticamente en una forma sensible con respecto al tipo de datos.

cin y **cout**: **cin** es el flujo de entrada estándar que normalmente es el teclado y **cout** es el flujo de salida estándar que por lo general es la pantalla. Para utilizar cualquiera de estos flujos debe incluirse **iostream** que es el archivo de encabezado del flujo de **entrada/salida.**

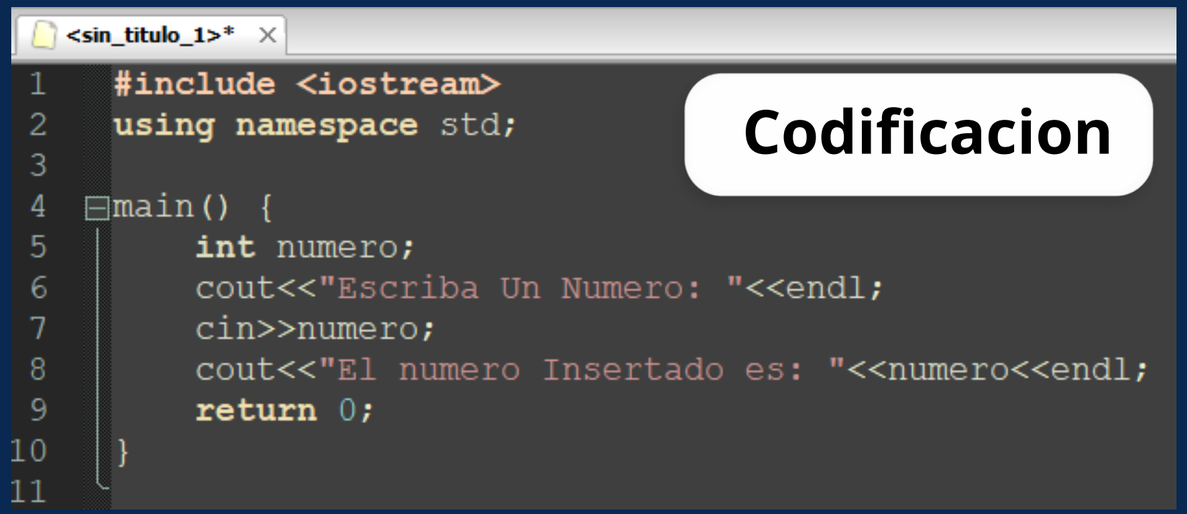
E/S DE DATOS EN C++

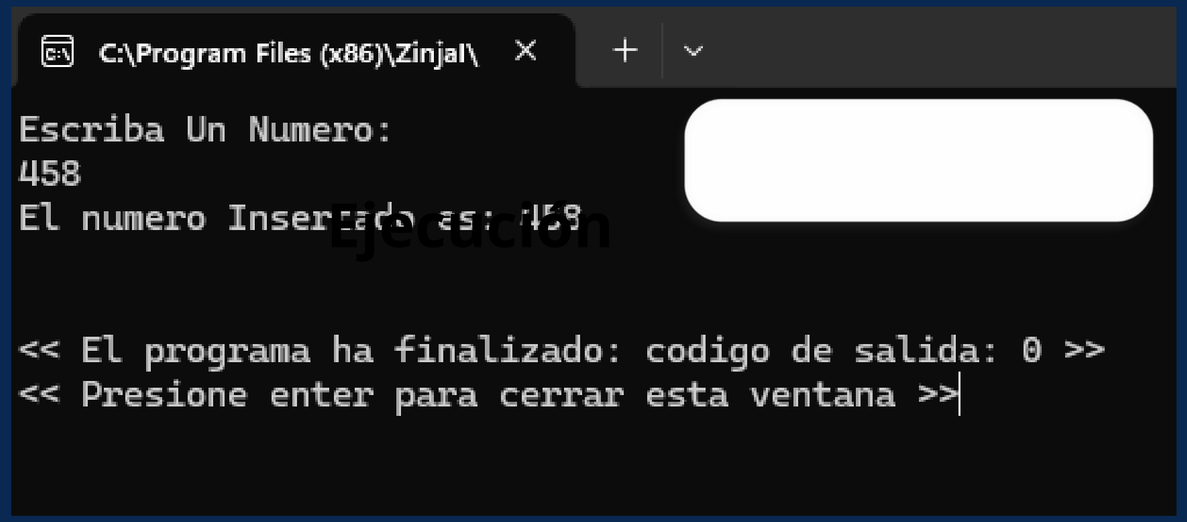
En **C++** se puede usar el flujo **cout** para imprimir información en la pantalla, utilizando el operador de salida de flujo “**<<**” .
Se puede **mostrar**: una cadena de texto, el contenido de una variable, el resultado de una expresión aritmética, etc.
La constante **endl** imprime un retorno de carro. También puede utilizarse “**\n**”.
Ejemplos:

cout<<“¡Hola Mundo…!”;
cout<<“El valor de x es ”<<endl;

El flujo de entrada **cin** permite introducir datos desde el teclado y almacenarlos en variables previamente declaradas.
Ejemplo:

int main() { int x;
cout<< “Ingrese un numero entero: “;
cin >> x;
cout<< “El numero que ingresaste es: “;
cout<< x << endl; }





ASIGNACIÓN DE VARIABLES Y EXPRESIONES ARITMÉTICAS

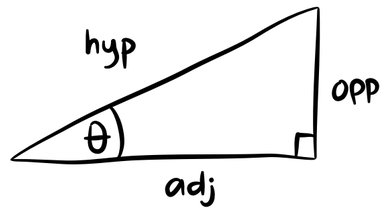
Para asignar un valor a una variable se utiliza el operador “=”.
Es posible asignar valores al momento de declarar las variables;

```
5       int var1 = 1;
6       char var2 = 'a';
7       float var3 = 13.5;
8       double var4 = 11.3;
9       bool var5 = true;
10
11      return 0;
```

Una operación aritmética es un proceso matemático básico que involucra números y operadores aritméticos. Las operaciones aritméticas forman la base de las matemáticas y son fundamentales en la programación, así como en la resolución de problemas numéricos.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

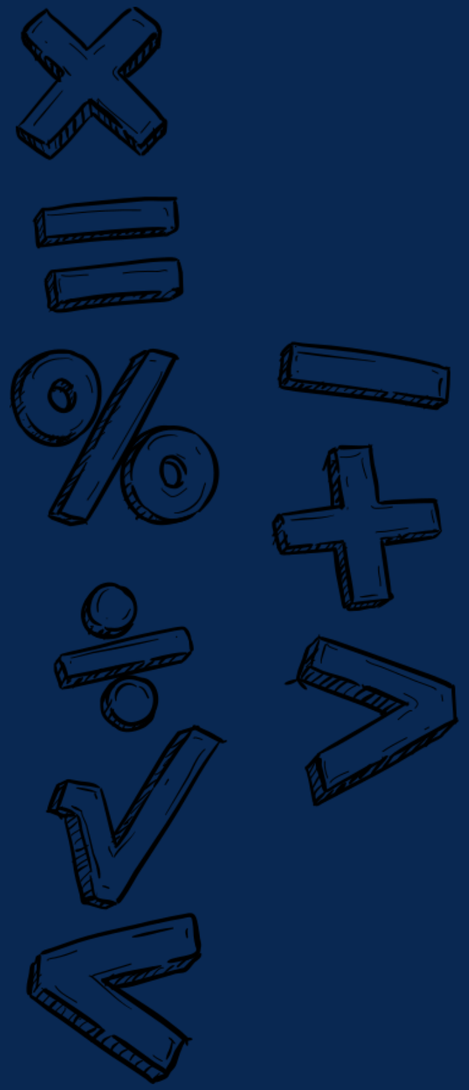
$$y = mx + b$$



$$ax + by = c \qquad \sin(\theta) = \frac{opp}{hyp}$$

OPERADORES ARITMETICOS EN C++

Tipo	Operadores
Asignación	=
Aritmético	Potencia: ***powf(x,y)*** (librería *math.h*); *x, y* valores numéricos Producto: * Cociente: / Módulo o resto: % Sumar: + Diferencia: -
Alfanuméricos	Operaciones con cadenas (librería *string.h*) *strcat(s,t);* concatena *t* al final de *s*. *strcmp(s,t);* compara *s* y *t*, retornando negativo, cero, o positivo para: *s<t, s==t, s > t.* *strcpy(s,t);* copia *t* en *s*. *strlen(s);* retorna la longitud de *s*. donde *s* y *t* son variables de tipo cadena.
Lógicos	Negación (NO, NOT): ! Conjunción (Y, AND): && Disyunción (O, OR): ||
Relacionales	Igual: == Distinto: != Mayor: > Mayor o igual: >= Menor: < Menor o igual: <=



OPERADORES ARITMÉTICOS

En C/C++ se pueden formar expresiones aritméticas con los operadores comunes: **+ (suma), - (resta), * (multiplicación), / (división), % (resto)**

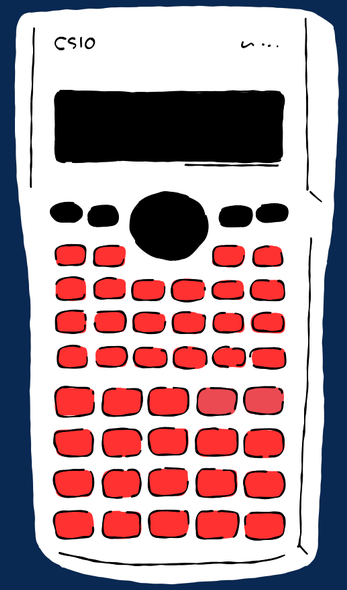
Debe tenerse siempre en cuenta que los operadores actúan de acuerdo con el tipo de dato de los operandos involucrados.

Ejemplo:

x = 5 / 2; //el resultado es 2

x = 5.0 / 2; //el resultado es 2.5

C/C++ no cuenta con un operador de potencia, pero se puede utilizar la función pow() de la librería math.h



OPERADORES INCREMENTALES

Los operadores **incrementales** y **decrementales** en C++ son operadores que se utilizan para **aumentar** o **disminuir** el valor de una variable en una unidad. Estos operadores son muy comunes y se utilizan a menudo en estructuras de control como bucles for, while, etc. Veamos cómo funcionan:

Operadores Incrementales

- Operador ++: Este operador incrementa el valor de la variable en 1.

Operadores Decrementales

- Operador --: Este operador disminuye el valor de la variable en 1.

++

```
int x = 5;
int y;

// Forma prefija
y = ++x; // x se incrementa a 6, y toma el valor de x, por lo tanto y = 6

// Forma sufija
y = x++; // y toma el valor de x (que es 6), luego x se incrementa a 7
```

--

```
int x = 5;
int y;

// Forma prefija
y = --x; // x se decrementa a 4, y toma el valor de x, por lo tanto y = 4

// Forma sufija
y = x--; // y toma el valor de x (que es 4), luego x se decrementa a 3
```

OPERADORES CON CADENAS

Para los operadores con cadenas es necesario incluir la Librería:

#include <string.h>

Lo cual nos permitirá hacer operaciones con cadenas.

strcat(Variable1,Varialbe2)

Concatena Variable1 y Variable2.

strcmp(Variable1, Variable2)

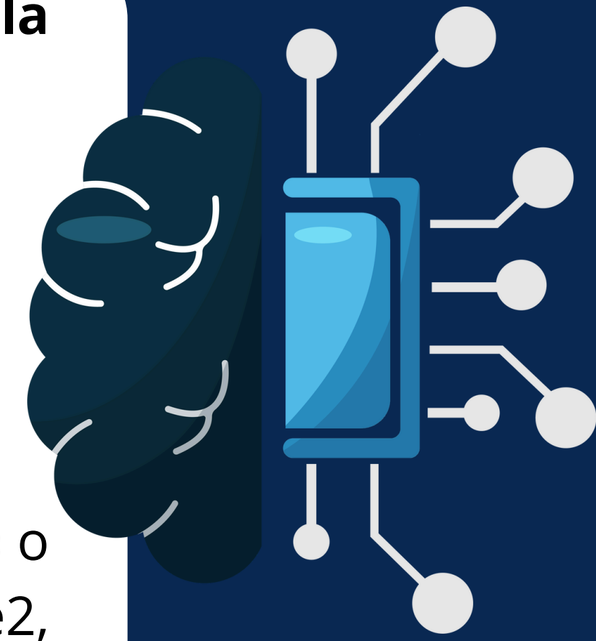
Compara Variable1 y Variable2 retornando Negativo(cero) o positivo para Variable1<Variable2, Variable1==Variable2, Variable1>Variable2.

strcpy(Variable1, Variable2)

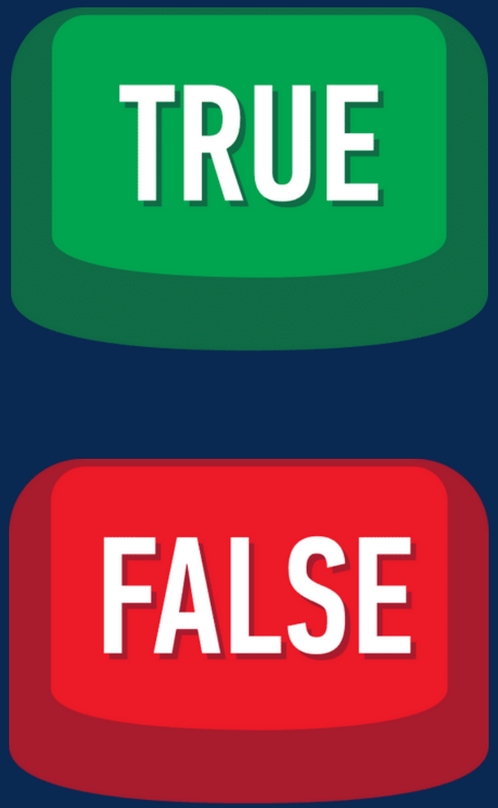
Copia el contenido la cadena de Variable2 en Variable.

strlen(Variable)

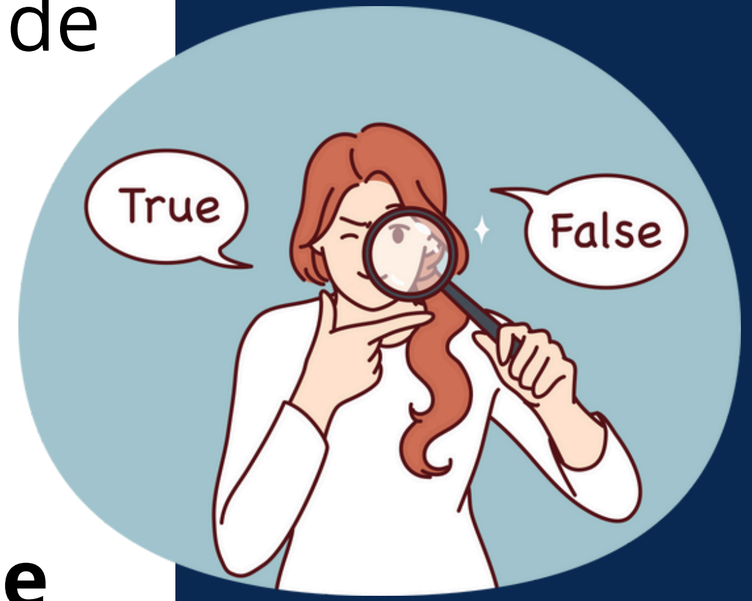
Retorna la Longitud de la Variable



EXPRESIONES BOOLEANAS



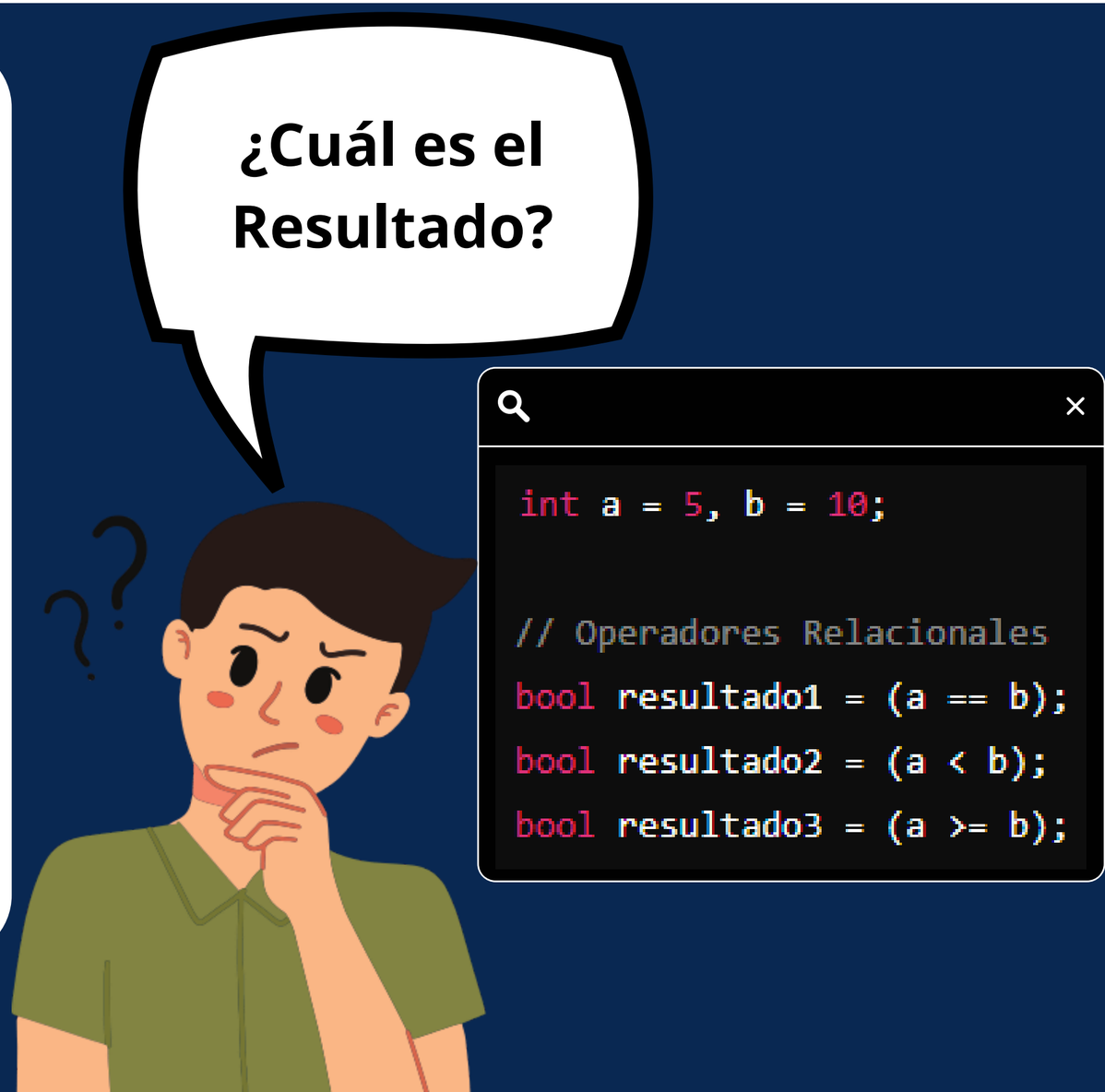
Una expresión booleana es una expresión que evalúa a un valor de tipo booleano, es decir, a **true** (verdadero) o **false** (falso). Estas expresiones son esenciales en estructuras de control como ‘if’, ‘while’, ‘for’, entre otras, **donde se necesita evaluar una condición.**



OPERADORES DE COMPARACIÓN

Una manera de formar **expresiones booleanas** es utilizando los operadores de comparación:

a == b	igualdad
a < b	menor a
a > b	mayor a
a != b	distinto de
a <= b	menor o igual a
a >= b	mayor o igual a



OPERADORES BOOLEANOS

Las expresiones booleanas pueden combinarse mediante los operadores booleanos clásicos:

- **&&** Conjunción (AND)
- **||** Disyunción (OR)
- **!** Negación (NOT)

&& : AND lógico. Devuelve **true si ambas expresiones son verdaderas.**

|| : OR lógico. Devuelve **true si al menos una de las expresiones es verdadera.**

! : NOT lógico. **Invierte el valor de la expresión**, es decir, **true** se convierte en **false** y viceversa.

Si a=7 y b=5 ¿Qué obtenemos?

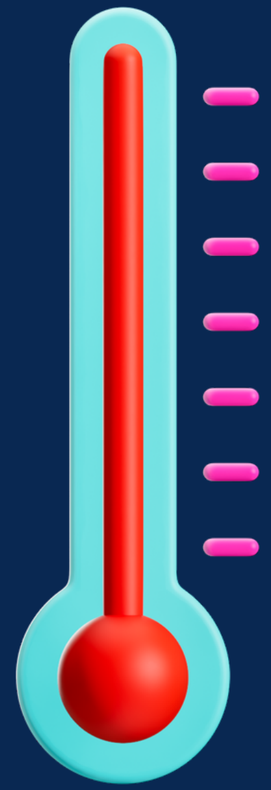
```
// Operadores Lógicos
bool resultado4 = (a < b && b > 0);
bool resultado5 = (a > b || b > 0);
bool resultado6 = !(a < b);
```

EJEMPLO 1

Eres un ingeniero industrial que trabaja en una planta de producción, y necesitas diseñar un sistema simple **para clasificar la temperatura del ambiente en diferentes zonas de la planta**. Para esto, debes escribir un programa que tome la temperatura actual (en grados Celsius) como entrada y, según el valor ingresado, clasifique la temperatura en una de las siguientes categorías:

- **Frío**: si la temperatura es menor a 15 grados.
- **Cálido**: si la temperatura está entre 15 y 25 grados (incluidos ambos).
- **Caliente**: si la temperatura es mayor a 25 grados.

Tu objetivo es crear un programa en C++ utilizando sentencias if para tomar la temperatura ingresada y mostrar el estado del clima en la planta de producción.



EJEMPLO 2

Como Ingeniero Industrial, **necesitas crear un sistema simple para identificar y verificar los nombres de las distintas áreas de trabajo de la planta**. El programa debe pedirle al usuario que ingrese el nombre de un área, y luego validar si el nombre ingresado coincide con alguno de los tres sectores más importantes de la planta:

- **Producción**
- **Logística**
- **Mantenimiento**

Si el nombre ingresado coincide con alguno de estos sectores, el programa debe confirmar su existencia en la planta. Si no coincide, el programa debe notificar que el área no existe.

EJEMPLO 3

Un Ingeniero Industrial necesita automatizar la verificación de ciertas condiciones en una máquina para saber si puede comenzar la producción. La máquina solo puede funcionar si todas las siguientes condiciones se cumplen:

1. El nivel de energía es suficiente (más de 50%).
2. La temperatura está dentro del rango permitido (entre 20°C y 80°C).
3. No hay fallas reportadas.

El programa debe verificar estas tres condiciones usando operadores lógicos.

¿PREGUNTAS?