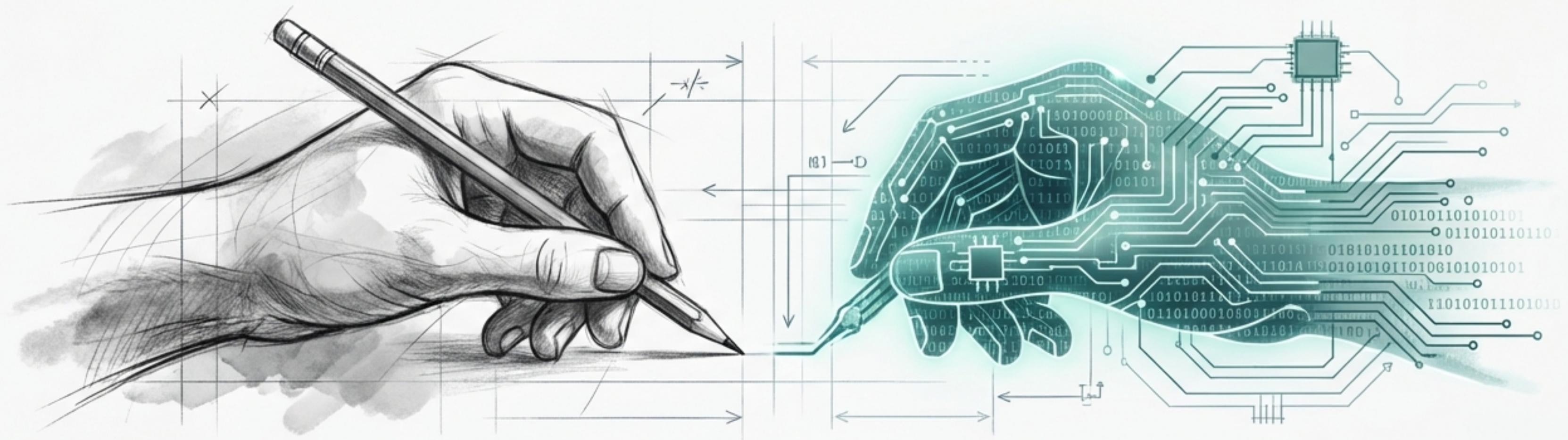


# El Arte de la Programación

Del Pensamiento Humano a la Acción de la Máquina

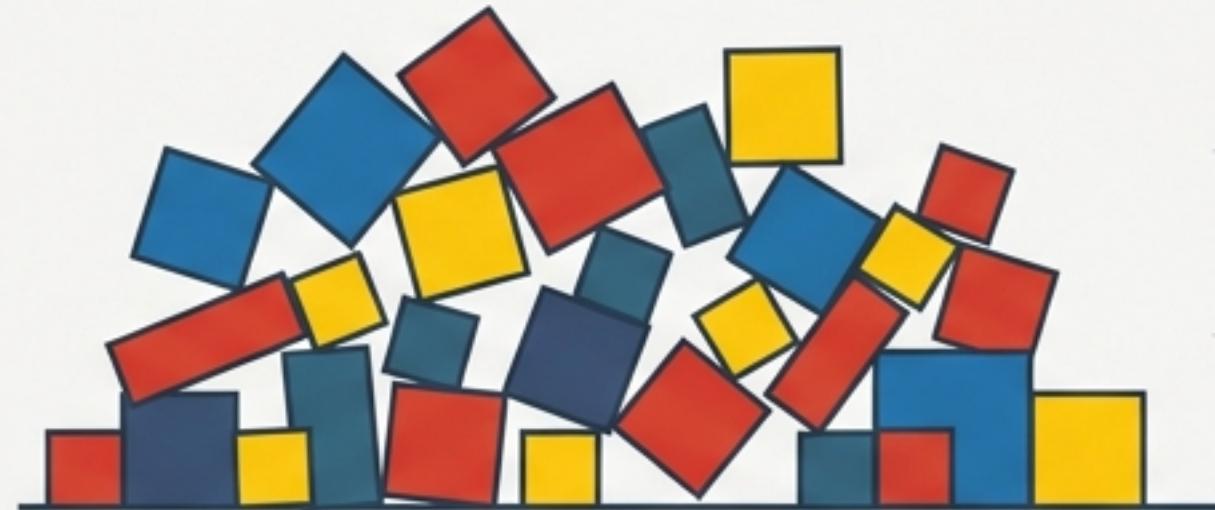


La programación es el nexo de unión entre el problema humano y la solución computacional.

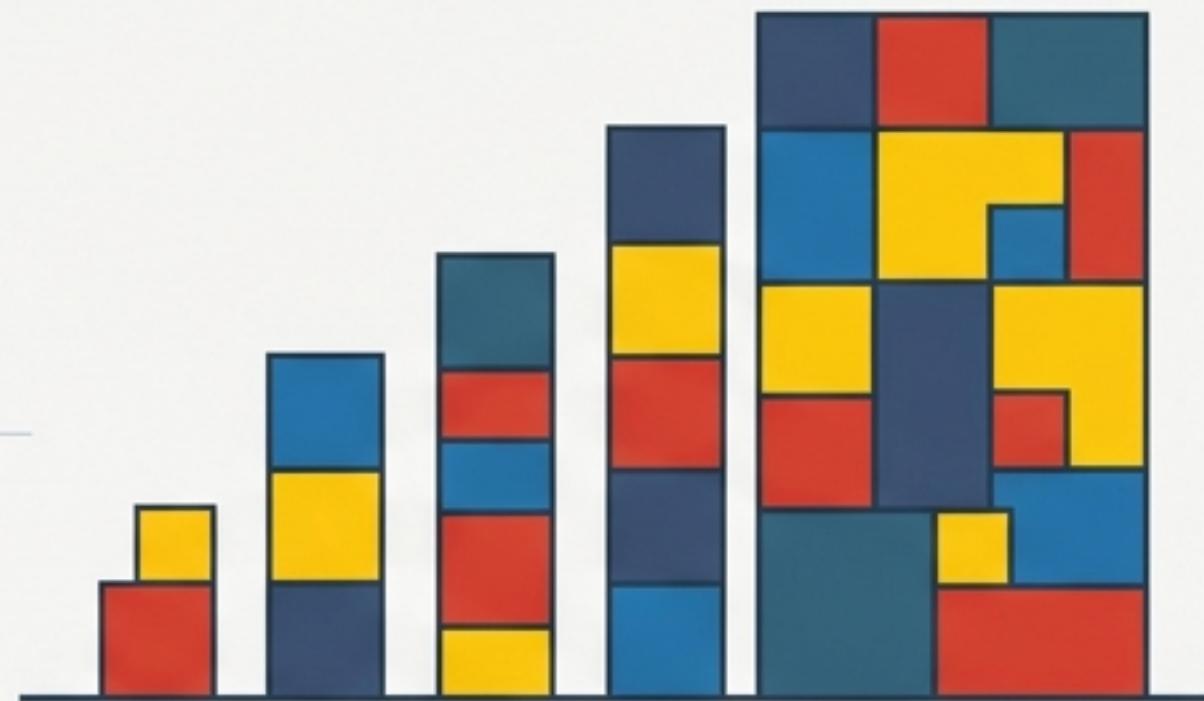
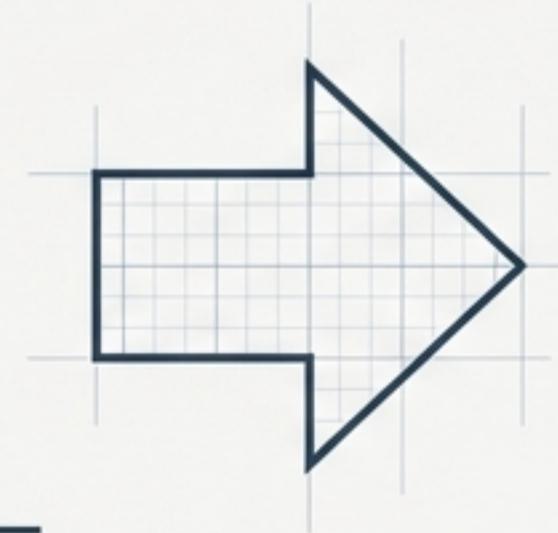
No es solo escribir código; es construir un modelo de solución preciso para que una máquina procese información y ejecute tareas complejas.

Bienvenidos al viaje del traductor: convertir ideas abstractas en instrucciones exactas.

# Datos vs. Información: Construyendo Valor



Datos



Información

**Datos:** Hechos objetivos crudos (números, palabras) sin contexto.

**Información:** El resultado de procesar, interpretar e interrelacionar esos datos para la toma de decisiones.

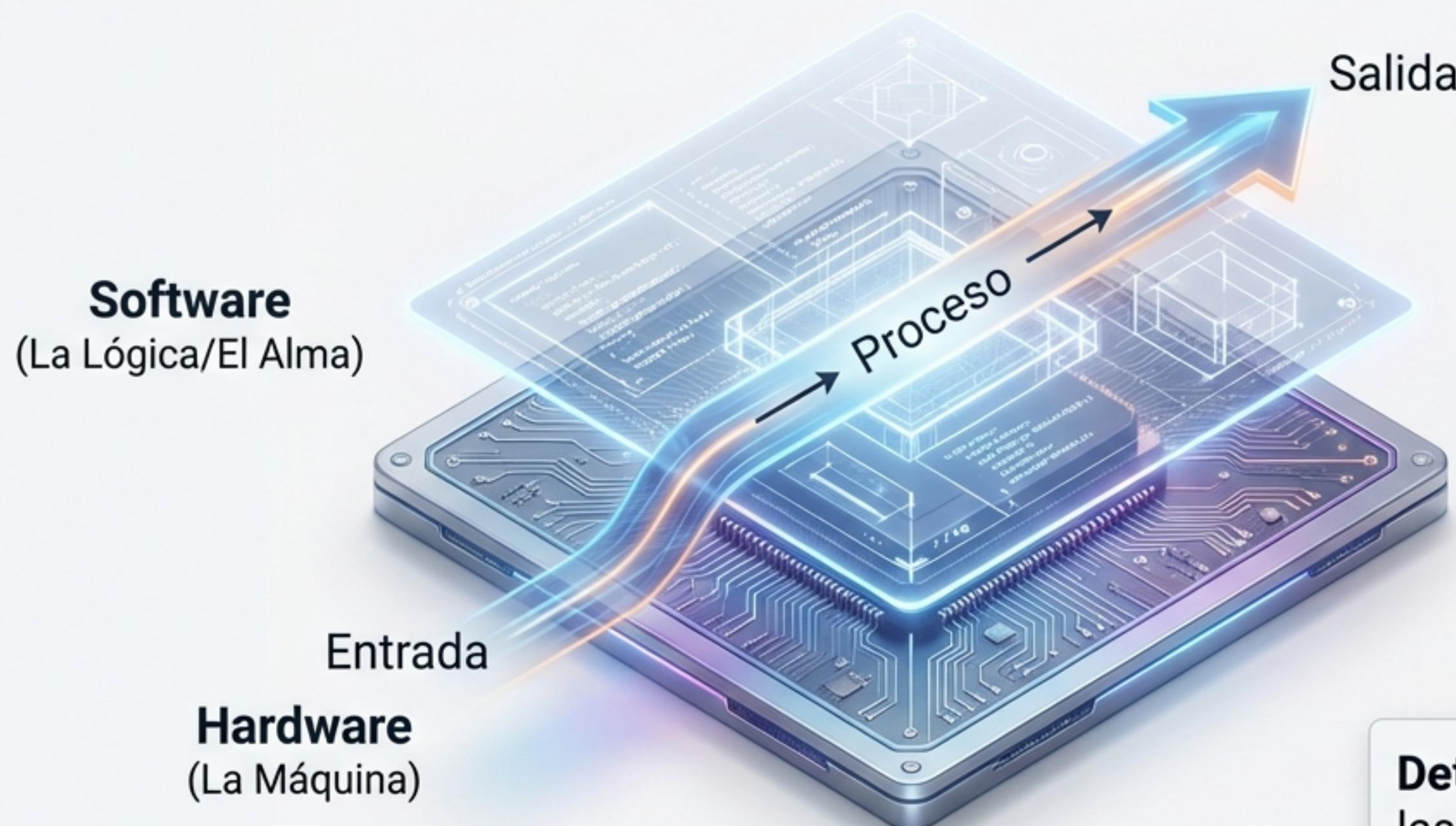
## Caso de Estudio: Inventario

**Los Datos:** "Smartphone Galaxy S23", "Precio: 999€",  
"Ventas: 20", "Stock: 30".

**La Información:** "El producto líder es el Galaxy S23. Es el más vendido y cuenta con el mayor stock disponible."

La informática automatiza este paso de datos crudos a información útil.

# El Motor de Procesamiento



1. **Hardware:** Elementos físicos y tangibles.
2. **Software:** La lógica intangible e instrucciones. Actualmente, a menudo más relevante que el hardware.
3. **Ciclo de Proceso:**
  - **Entrada:** Datos introducidos.
  - **Unidad de Proceso:** Transformación de datos.
  - **Salida:** Resultado final.

**Determinismo:** Ante la misma entrada y las mismas transformaciones, el resultado debe ser siempre el mismo.

# El Algoritmo: La Receta de la Lógica

Receta para Resolver Problemas

1. Entender el problema.
2. Desglosar en pasos.
3. Ejecutar en orden.
4. Verificar resultado.
5. Finalizar.

Un conjunto finito y ordenado de instrucciones bien definidas para resolver un problema.

```
function resolverProblema() {  
    entrada = getDatos();  
    procesamiento = [];  
    for (paso of pasos) {  
        procesamiento.push(ejecutar(paso));  
    }  
    salida = verificar(procesamiento);  
    return salida;  
}
```

## 5 Características del Algoritmo

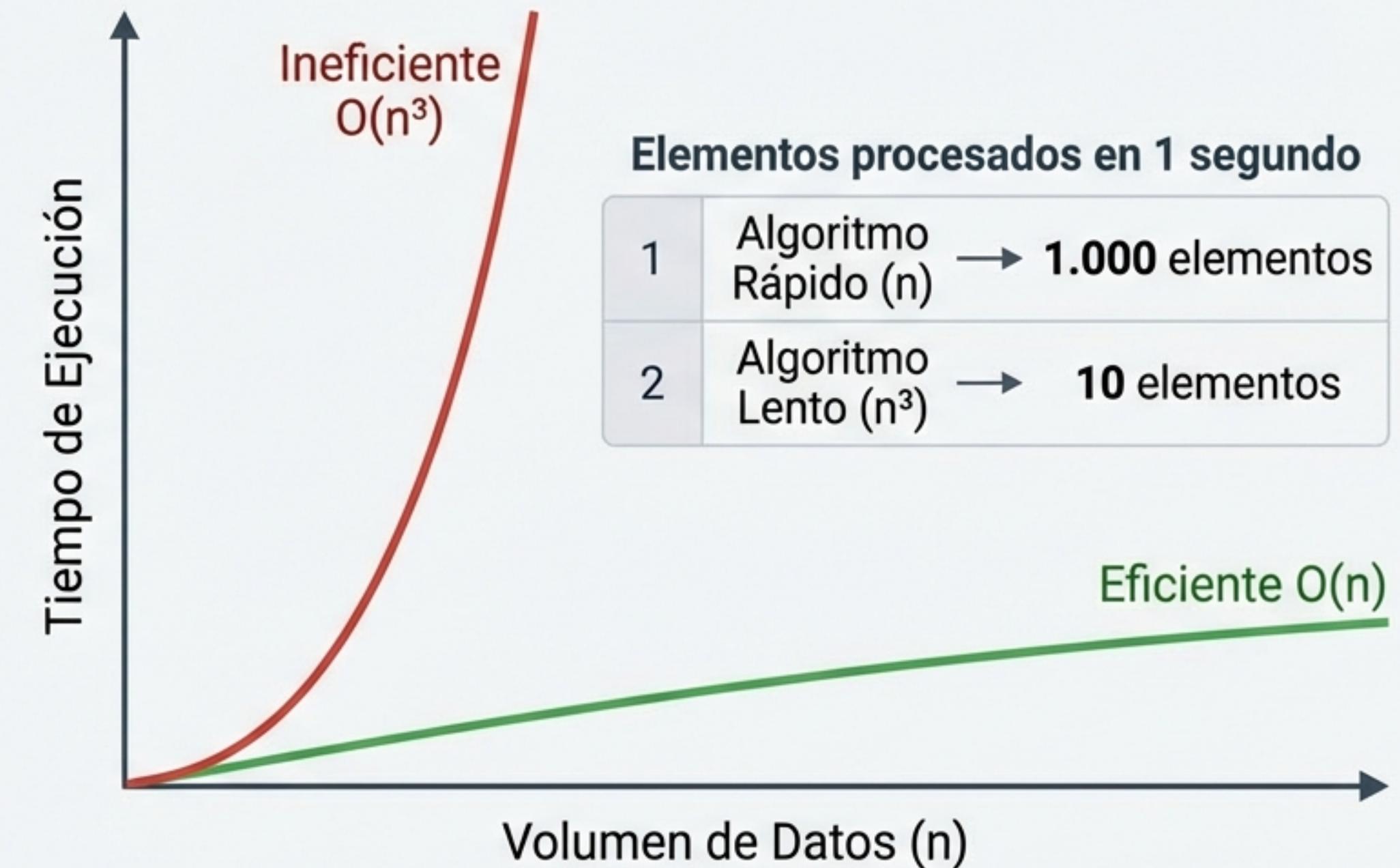
- ✓ **Preciso:** Orden exacto de cada paso.
- ✓ **Definido:** Sin ambigüedades.
- ✓ **Determinista:** Mismos datos = Mismo resultado.
- ✓ **Finito:** Número limitado de pasos.
- ✓ **Independiente:** Sirve para cualquier lenguaje.

**Algoritmo vs. Programa:** El algoritmo es la solución abstracta; el programa es la traducción para la máquina.

# Midiendo el Éxito: Eficacia vs. Eficiencia

**Eficacia:** ¿Funciona?  
(Cumple los requisitos).

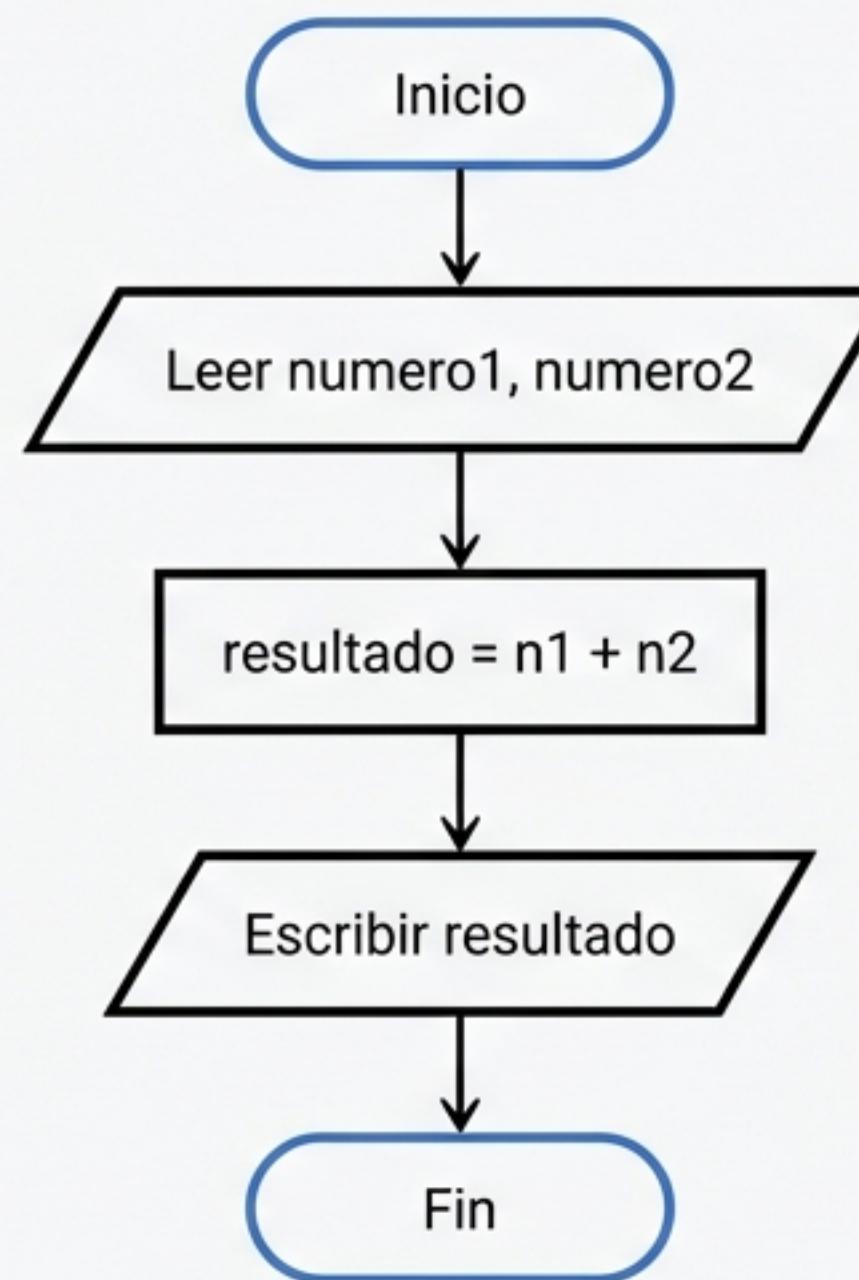
**Eficiencia:** ¿Funciona bien? (Usa el mínimo tiempo y memoria).



La complejidad ( $O(n)$ ) determina si un sistema es viable o si colapsará bajo carga.

# Representación: Visualizando la Solución

## Diagrama de Flujo (Gráfico)



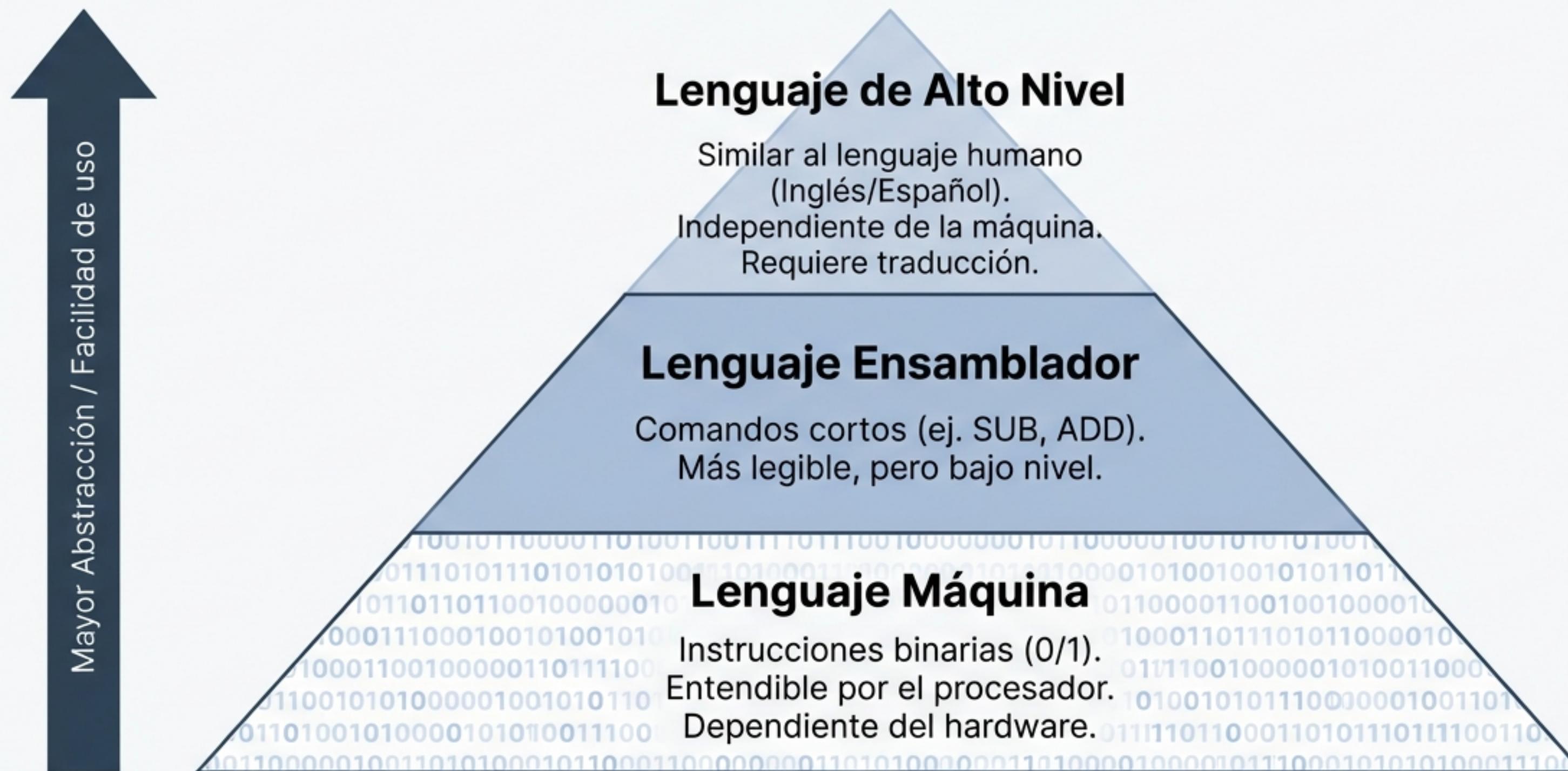
## Pseudocódigo (Textual)

```
Inicio
    // Pedir al usuario números
    Escribir 'Ingrese el primer número:'
    Leer numero1
    Escribir 'Ingrese el segundo número:'
    Leer numero2

    // Sumar
    resultado = numero1 + numero2

    // Mostrar
    Escribir 'La suma es:', resultado
Fin
```

# La Barrera del Idioma: Niveles de Abstracción



# La Traducción: Compiladores e Intérpretes

## Compilador (ej. C, Pascal)



Genera un archivo independiente. Más eficiente.

## Intérprete (ej. Python, JS)



Ejecuta instrucción por instrucción. Más lento, fácil de depurar.

# La Gramática del Código

## Sintaxis (La Forma)

Reglas gramaticales de escritura.

```
int a = 5;  
int b = 10;  
int c = a + b
```

```
int result = a * b
```

**Error:** Faltar un punto y coma. El programa no compila.

## Semántica (El Significado)

El sentido o lógica del código.

```
int a = 10;  
int b = 20;  
int suma = a + b;
```

```
int a = 10;  
int b = 20;  
int suma = a - b;
```

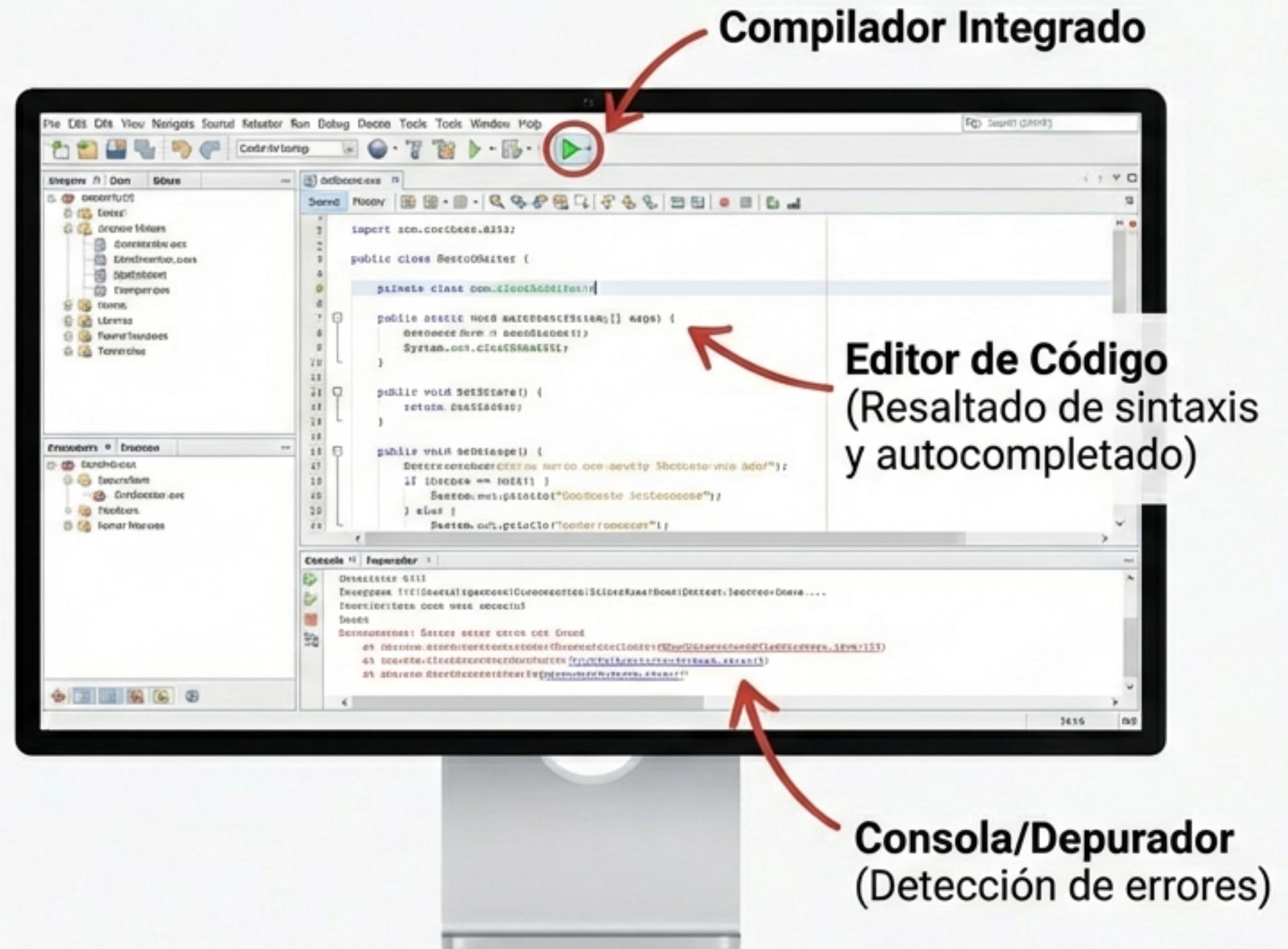
**Error:** Restar en lugar de sumar. El programa funciona, pero el resultado es incorrecto.

**Estilos:** Imperativo (Cómo hacerlo paso a paso) vs. Declarativo (Qué resultado se quiere).

# El Ciclo de Vida del Software (SDLC)



# El Entorno de Desarrollo (IDE)



## ¿Por qué usar un IDE?

- Unifica herramientas.
- Aumenta productividad.
- Detecta errores en tiempo real.

Ejemplo: NetBeans.

# Síntesis: El Ecosistema de Programación



# Su Turno para Construir



- La programación es un proceso de comunicación con una máquina.
- No existe una regla única; la única forma de dominar el arte es practicando.
- El código es la herramienta; su lógica es la solución.

**Valide siempre la Finitud y el Determinismo en sus soluciones.**