

# Lenguaje de programación

## Lenguajes de Programación

- Los lenguajes de programación describen cálculos a humanos y máquinas.
- Todo software está escrito en un lenguaje de programación.
- Para ejecutarse, un programa debe traducirse a un formato comprensible por la máquina.

## Evolución de los Lenguajes de Programación

### Lenguajes de bajo nivel:

- Lenguaje máquina: usa 0s y 1s.
- Ensamblador: usa instrucciones mnemotécnicas.

### Lenguajes de alto nivel:

- Tercera generación: Fortran, C, Java.
- Cuarta generación: SQL, NOMAD (para aplicaciones específicas).
- Quinta generación: Prolog (basados en lógica).

### Clasificación por paradigma:

- Imperativo: Especifica cómo hacer un cálculo (C, Java).
- Declarativo: Especifica qué calcular (Prolog, Haskell).
- Orientado a Objetos: Se basa en objetos que interactúan (C++, Java, Ruby).
- Lenguajes de secuencias de comandos: Interpretados y flexibles (Python, PHP).

## Procesadores de Lenguaje

- **Compilador:** Traduce el programa fuente a un lenguaje destino ejecutable.
- **Intérprete:** Ejecuta directamente el código fuente sin generar un programa intermedio.
- **Ejemplo:** Java usa una combinación de compilación e interpretación (Bytecodes y Máquina Virtual Java).

## Relación entre Lenguajes y Compiladores

- Los lenguajes de programación influyen en la evolución de los compiladores.
- Los compiladores deben adaptarse a nuevas características de los lenguajes.
- Son clave para optimizar el rendimiento de los programas.
- Un buen compilador puede mejorar el uso de arquitecturas de alto rendimiento.
- Crear compiladores requiere conocimientos de teoría y práctica en ciencias computacionales.

## Compilador

- Son sistemas de software que traducen un programa de un lenguaje fuente a un lenguaje destino.

## Fases de un compilador

- **Análisis léxico:** Divide el código en tokens.
- **Análisis sintáctico:** Organiza los tokens en una estructura gramatical.
- **Análisis semántico:** Verifica la consistencia del código con las reglas del lenguaje.
- **Generación de código intermedio:** Crea una representación intermedia del programa.
- **Optimización de código:** Mejora la eficiencia del código.
- **Generación de código:** Traduce el programa intermedio a código ejecutable.
- **Tabla de símbolos:** Almacena información sobre variables y funciones del programa.