Ministerio de Producción
Presidencia de la Nación

Ministerio de Educación y Deportes

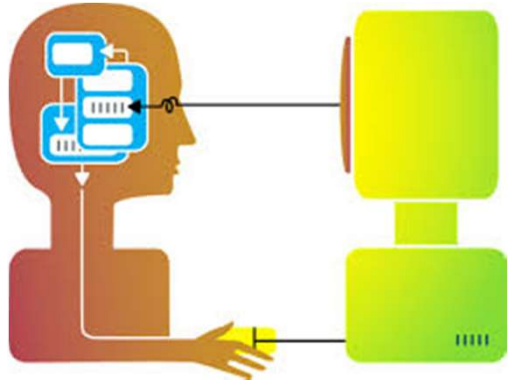
Subsecretaría de Servicios Tecnológicos y Productivos

Programa 111 mil
VOS PODÉS SER UNO.

LENGUAJES DE PROGRAMACIÓN
PARADIGMAS

Lenguajes de Programación

*Se refiere a todos los símbolos, caracteres y reglas de uso, que permiten a las personas **comunicarse** con las computadoras*

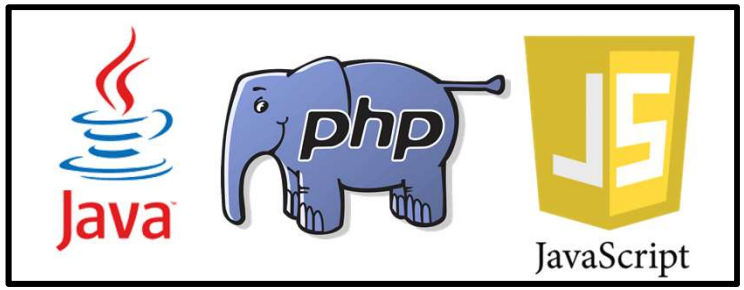


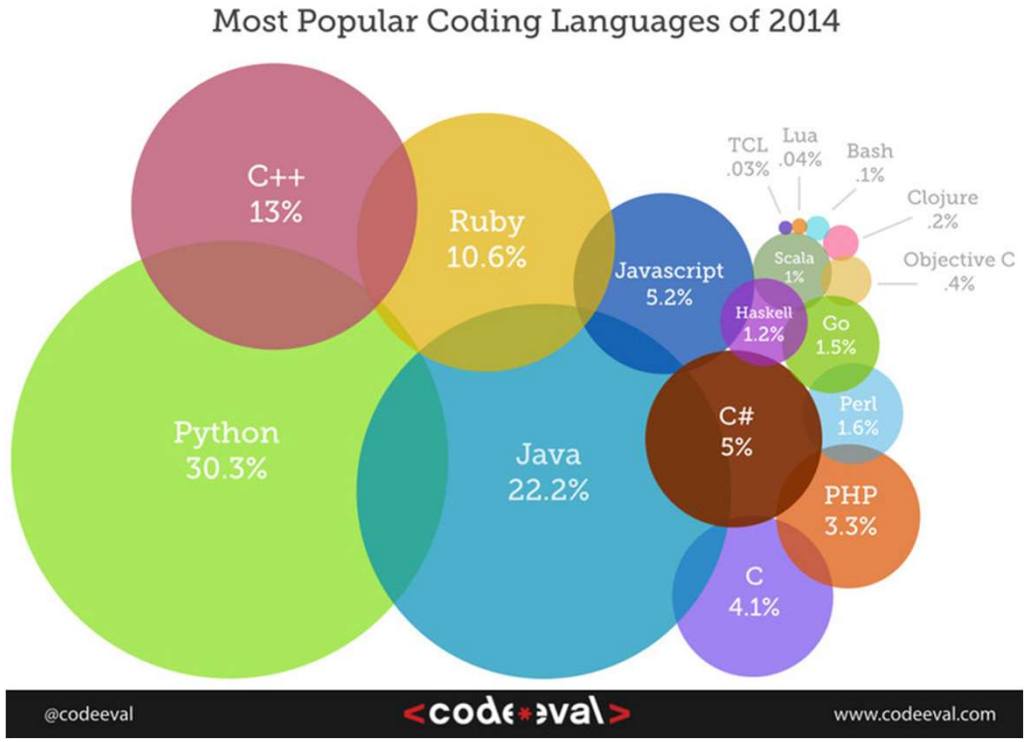
- Un **lenguaje de programación** es:
  - un sistema estructurado y diseñado para que las computadoras se entiendan entre si
  - y se entiendan con los humanos
  - un conjunto de acciones que la computadora debe ejecutar
  - diferentes normas para controlar como se comporta una maquina
  - se usan para crear **programas informáticos → productos de SW**

- **Programación**: proceso de diseño, codificación, depuración y prueba de código (para computadoras)

Ejemplos de Lenguajes

- Todos los lenguajes tienen instrucciones en las categorías: entrada-salida, cálculo, manipulación de texto, lógica, comparación, almacenamiento, y recuperación





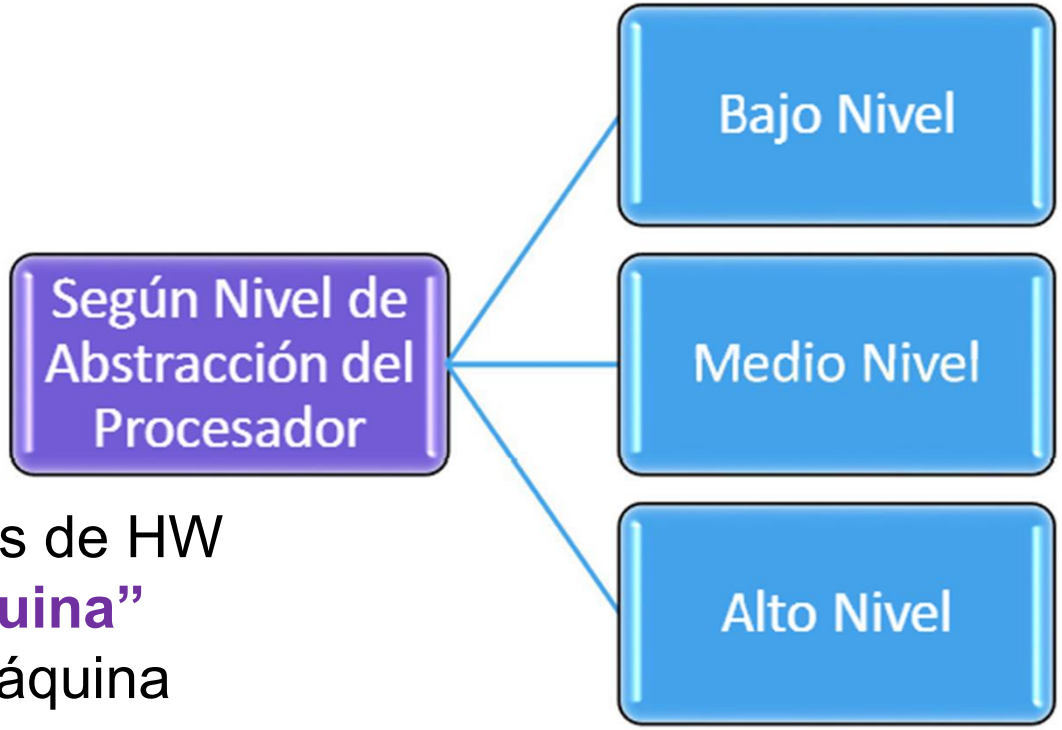
Tipos de Lenguajes

1. Según Nivel de Abstracción del Procesador

2. Según Paradigma de Programación

3. Según Forma de Ejecución

Nivel de Abstracción del Procesador - 1



Bajo nivel

- poca o ninguna abstracción del microprocesador
  → **Más cerca de la máquina**
- fácilmente trasladable a lenguaje máquina para drivers de HW
- también llamado **"código máquina"**
- Es código dependiente de la máquina (sin portabilidad)
  Exige mayor esfuerzo (cognitivo) de los programadores

Alto nivel

- Permiten expresar algoritmos de manera más adecuada para seres humanos → **Más cercano al "modo de pensar" del programador**

Nivel de Abstracción del Procesador - 2

Alto nivel (continuación)

- Requiere mayor conocimiento de programación para codificar
- Permite **mayor productividad** del programador
  (ej., para procesamiento de datos)

Medio nivel

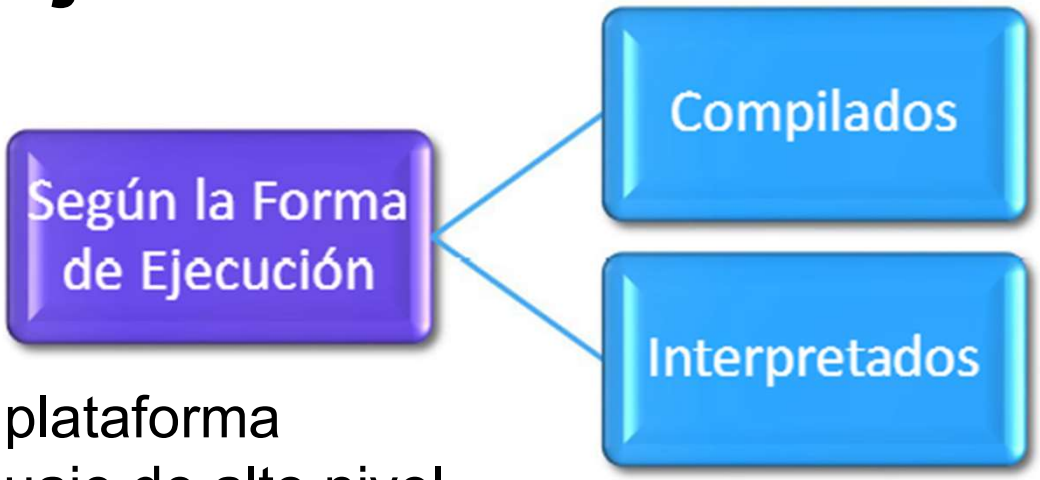
- Son lenguajes **híbridos** (ej., C)
- Si bien funciona con características de alto nivel, también permite ciertos manejos a bajo nivel

```
class Triangle {
  ...
  float surface()
    return b*h/2;
  }
```

```
LOAD r1,b
LOAD r2,h
MUL r1,r2
DIV r1,#2
RET
```

```
0001001001000101
0010010011101100
10101101001...
```

Forma de Ejecución



Compilados

- Un **compilador** traduce un programa escrito en un lenguaje a lenguaje máquina (binario), para una determinada plataforma
- El código fuente está en un lenguaje de alto nivel
- Ejemplos: C, C++, Objective-C, Fortran, Pascal, Visual Basic, **Java**.

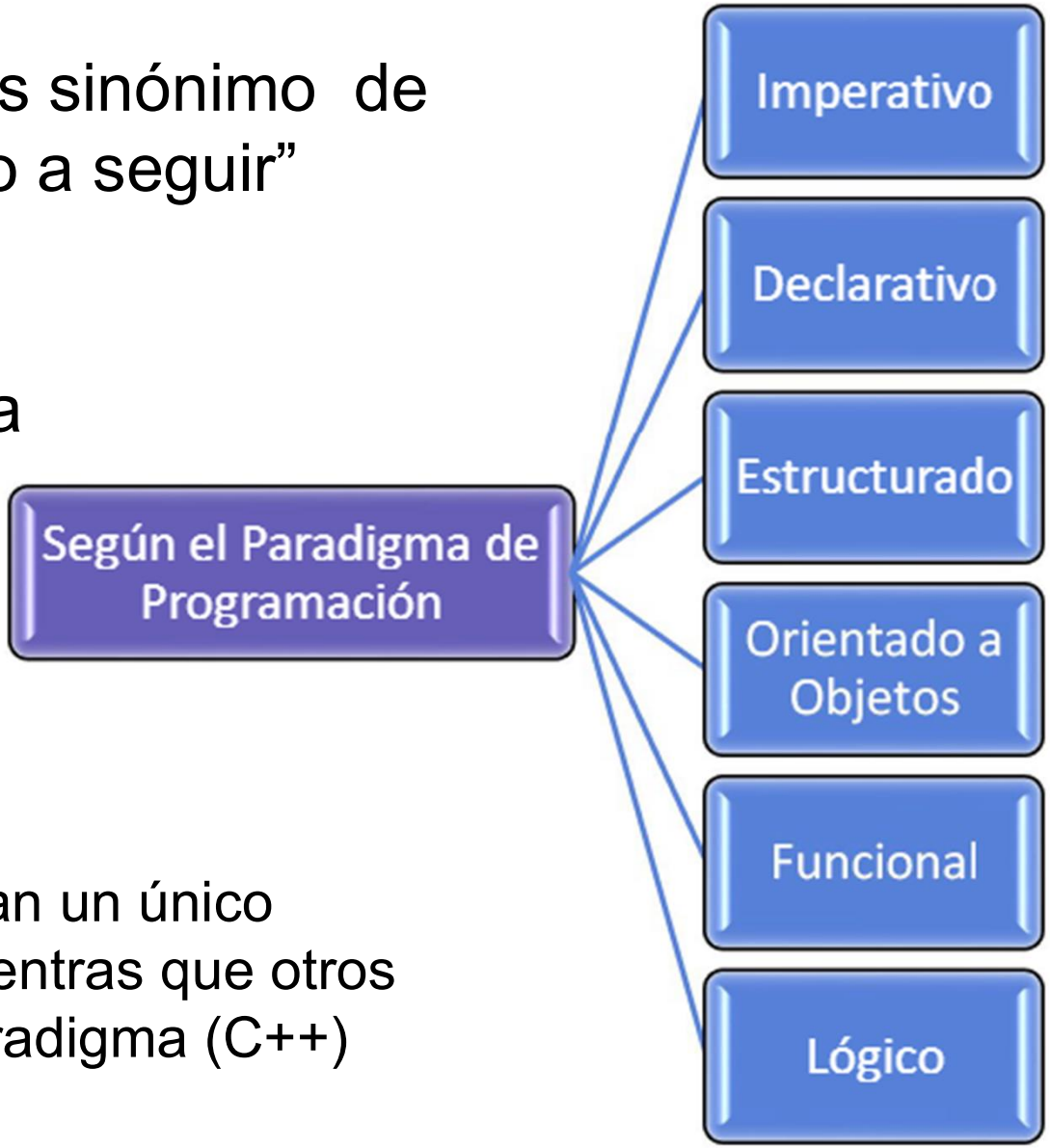
Interpretados

- El código fuente está escrito en un lenguaje de alto nivel, y es traducido por un **intérprete** a un lenguaje de más bajo nivel, en una modalidad **paso a paso** (instrucción por instrucción)
- La interpretación se suele realizar en tiempo de ejecución
- Ejemplos: Ruby, Python, PHP, JavaScript, Smalltalk, Matlab, **Java**.

Paradigmas de Programación - 1

- En términos generales, es sinónimo de "marco teórico" o "modelo a seguir"

- En computación
  Un paradigma es la forma de "entender el mundo"
  - **Visión y metodología para la construcción de un programa**
  - **Filosofía para diseñar, concepto de solución**
  - Algunos lenguajes soportan un único paradigma (Smalltalk), mientras que otros lenguajes son múltiple paradigma (C++)



Paradigmas de Programación - 2

1. Imperativo

Secuencia de instrucciones o comandos que cambian el estado de un programa -- ej., código máquina.
Incluye al paradigma procedural

2. Declarativo

No se basa en el cómo se hace (lograr un objetivo paso a paso), sino que se describe (declara) cómo es algo
Ejemplo: Prolog
Se describen las propiedades de la solución busca, y se deja "indeterminado" el algoritmo para encontrar esa solución.
Tiene desventajas en eficiencia, pero ventajas para problemas complejos (por ej., IA) o de prototipado de soluciones

Paradigmas de Programación - 3

3. Estructurado

La programación se divide en bloques (procedimientos y funciones), que pueden o no comunicarse entre si. La programación se controla con secuencia, selección e iteración.
Permite reutilizar código y otorga una mejor "comprensión" de la programación. Ejemplo: Pascal

4. Funcional

Concibe la computación como la evaluación de funciones matemáticas, y evita declarar y cambiar datos
Se basa en la aplicación de funciones y composición entre ellas (más que en los cambios de estados y la ejecución secuencial de comandos – como en el paradigma procedural)
Evita los "efectos secundarios" comunes de otros paradigmas
Ejemplo: Lisp, Haskell, Erlang

Paradigmas de Programación - 4

5. Lógico

Se basa en conceptos de lógica matemática y reglas
Trabaja con predicados que relacionan a los individuos involucrados
Permite deducción de la(s) posible(s) respuesta(s) de una consulta a través de un motor, para resolver problemas.
Ejemplo: Prolog

6. Orientado a Objetos

Intenta simular el "mundo real" y las soluciones a través del **modelado de objetos**, cada uno con características y funciones propias
Generalmente basado en clases y métodos.
Los objetos se comunicación mediante el intercambio de mensajes
Su principal ventaja es la reutilización de diseño
Ejemplo: Java, Smalltalk, Scala