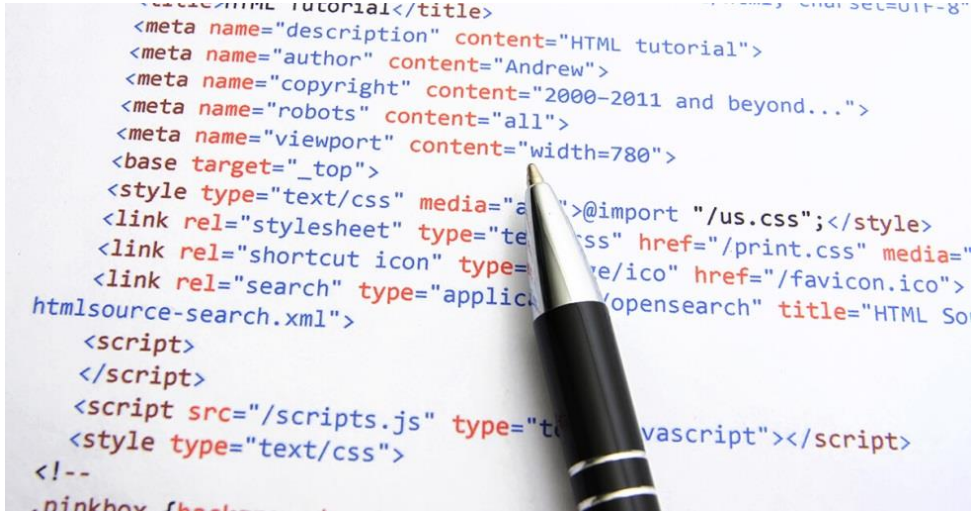
Interprete y Compilador

Elementos de Informática

Código Fuente

El código fuente es la secuencia de instrucciones que un programa transmite a una computadora para que pueda ejecutarse.

Dichas instrucciones son líneas de texto escritas en un lenguaje de programación cercano al programador respetando la semántica y la sintaxis del lenguaje .



Código Objeto

Es aquel programa que se encuentra en lenguaje máquina y que ya es ejecutable por esta.
Es el resultado de traducir el código fuente a un lenguaje comprensible por la compuatdora en ceros y unos.

Traductor

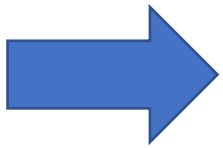
Lee un código fuente (Alto Nivel) y lo traduce a código objeto o lenguaje de máquina (Bajo Nivel).

```java
// class declaration
public class ProgrammingExample {

    // method declaration
    public void sayHello() {

        // method output
        System.out.println("Hello World!");
    }
}
```

Lenguaje Alto Nivel



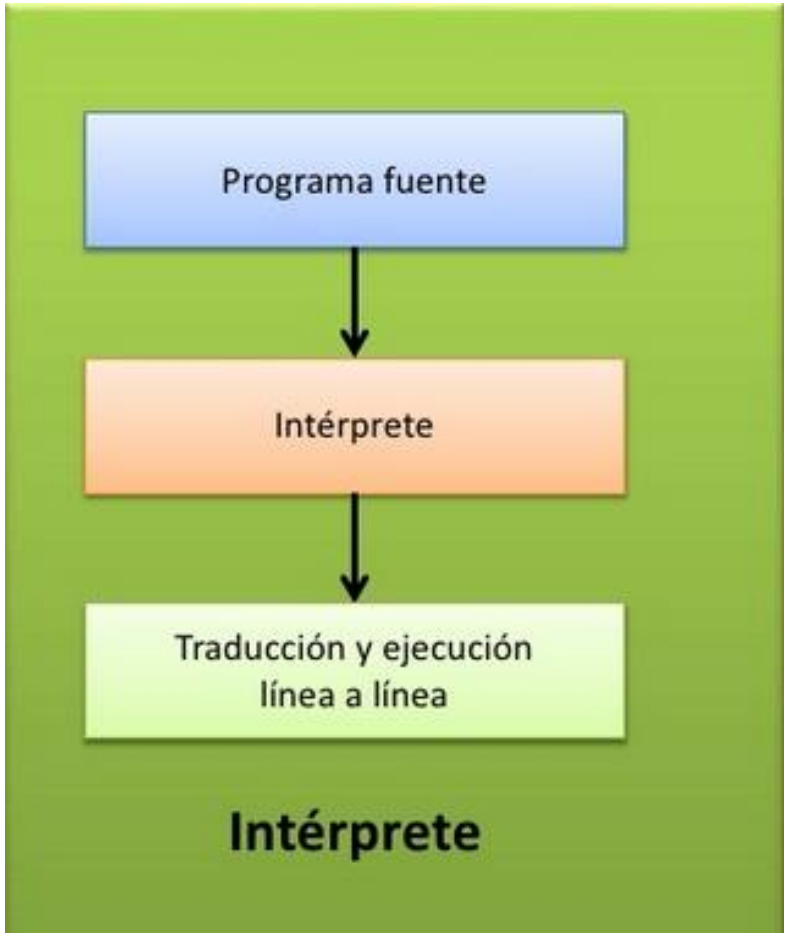
```
0010 0001 1010
1010 1011 1000
0110 1101 0001
0010 0001 1010
1010 1011 1000
0110 1101 0001
```

Lenguaje Máquina o Bajo Nivel

Tipos de Traductores

- **INTERPRETES.**
- **COMPILADORES**

Intérprete



Traductor de Lenguajes de Alto Nivel.

El Programa Fuente siempre aparece en su forma original.

Ejecuta el Programa línea x línea

La traducción se hace al momento de ejecutar cada instrucción

Intérpretes:
Principales Características

Requiere de un programa auxiliar (el intérprete)

Traduce y Ejecuta el Programa línea x línea.

No se graba un Código Objeto (ejecutable) para usarlo otra vez.

Es más Lento en Ejecución y mas Rápido en Diseño.

Intérprete: Ventajas

- Un intérprete necesita menos memoria.
- Permiten una mayor interactividad con el código en tiempo de desarrollo.
- El intérprete elimina la necesidad de realizar una “verificación completa” después de cada modificación del programa.
- Independencia de plataforma.

Intérprete: Desventajas

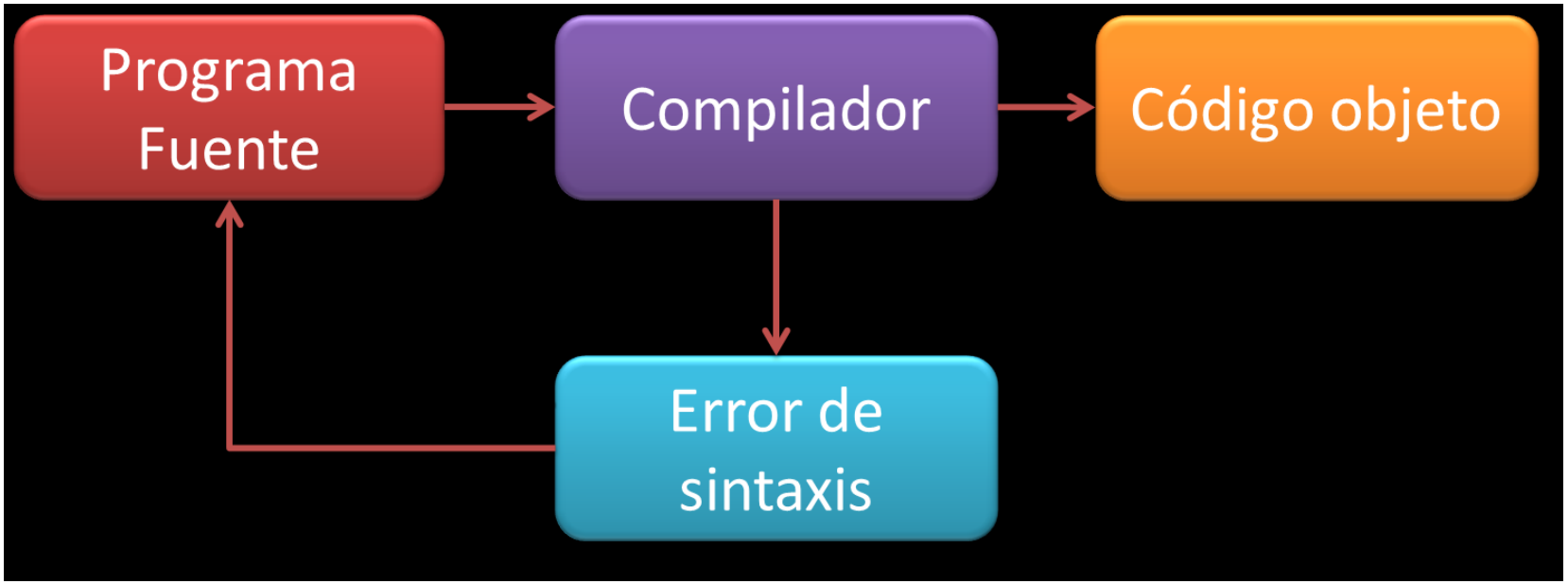
- Un código traducido por un intérprete no puede funcionar sin éste.
- Se Ejecuta n veces se interpreta n veces.
- No hay Independencia entre la etapa de traducción y ejecución.

Intérprete: Ejemplos

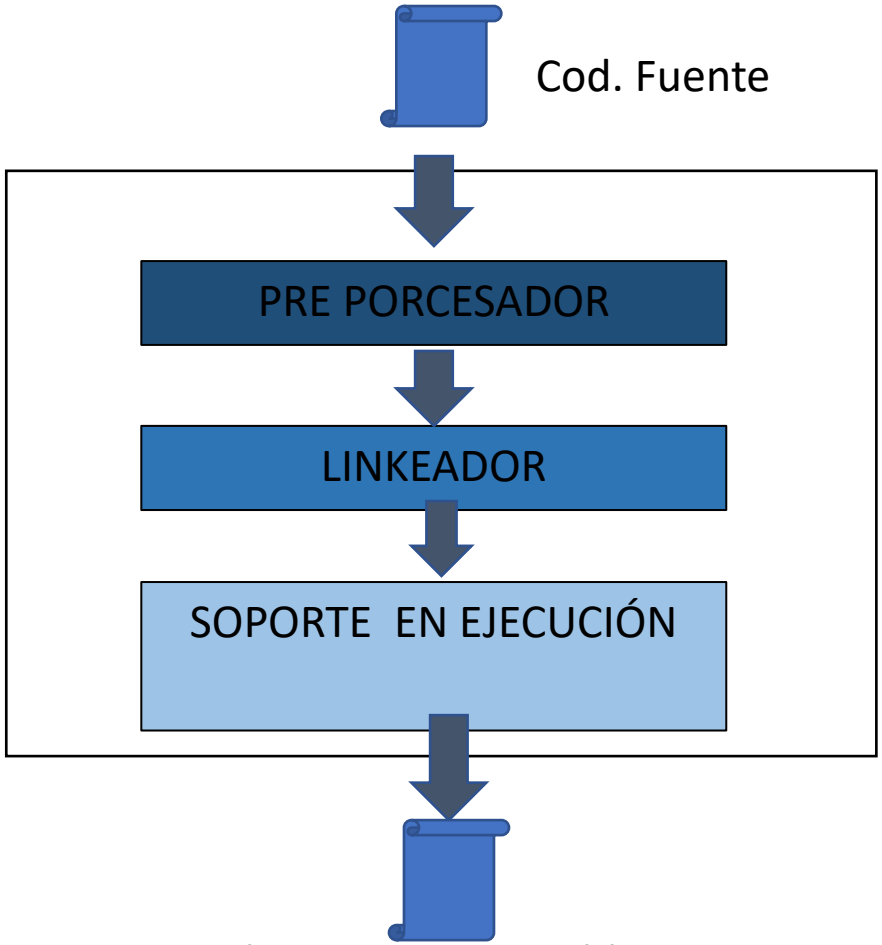
Lenguajes que utilizan Interpretes:

BASIC 4GL	Java Script
LOGO	LISP
PERL	PHP
HTML	PROLOG
PYTHON	RUBY

¿Qué es un compilador?



Composición de un compilador

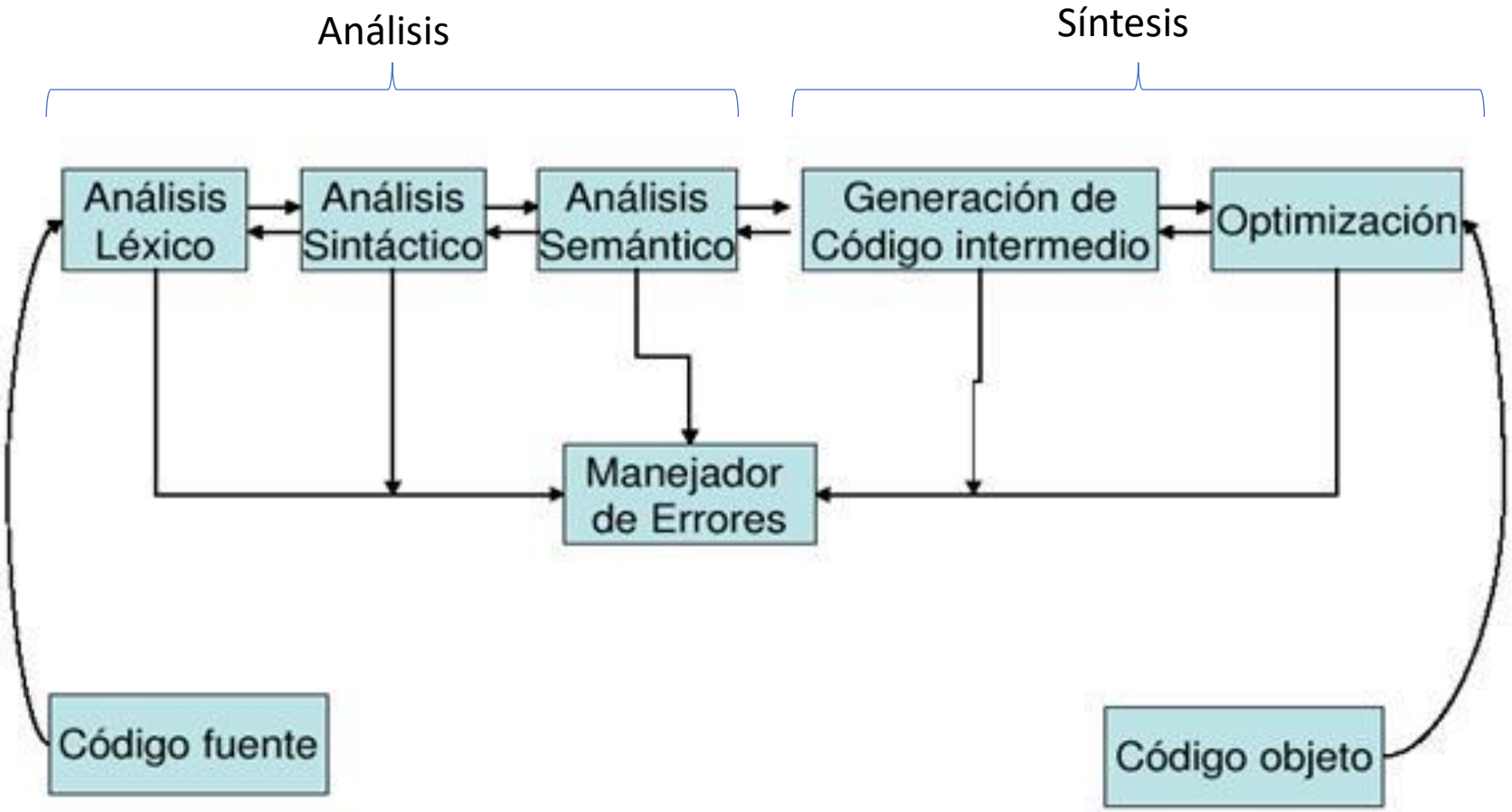


Pre procesador: En esta fase se realizan acciones tales como expansión de macros y la inclusión de ficheros

Linkeador: generalmente el resultado de la compilación del código fuente es una parte de programa que debe unirse a otras librerías que están compiladas y así crear el programa completo. El linkeador es el encargado de unificar diferente módulos.

Soporte en ejecución: Algunas de sus funciones son preparar la memoria al comienzo de la ejecución, gestionar la memoria o la pila durante la ejecución y preparar la finalización de la ejecución de una manera razonable.

Fases de un Compilador

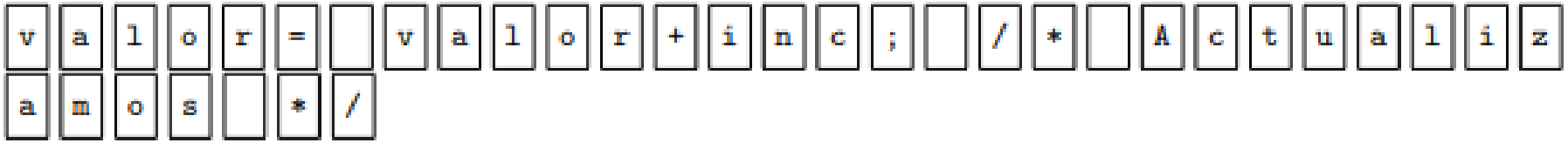


Compiladores: Tarea de Análisis

Análisis LEXICO**:** se encarga de la división de la entrada en componentes léxicos.

```
valor= valor+inc; /* Actualizamos */
```

Entrada



Salida

Compiladores: Tarea de Análisis

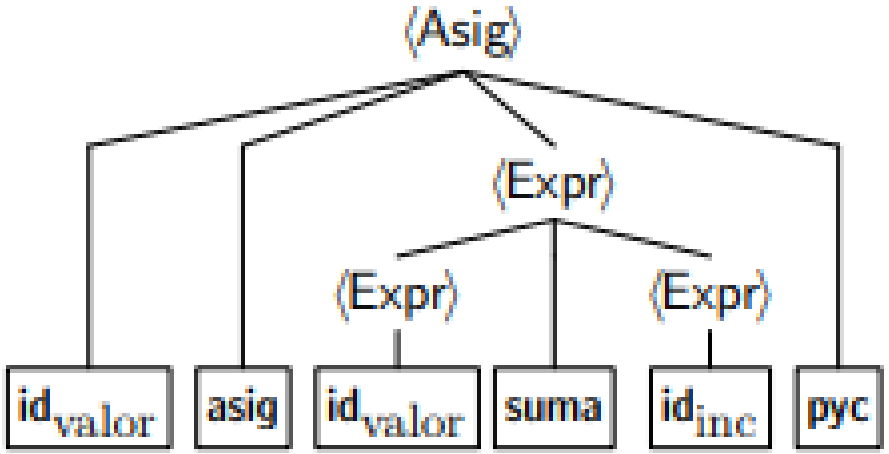
Análisis SINTÁCTICO: se encarga de encontrar las estructuras presentes en la entrada

⟨Expr⟩ → **id**

⟨Expr⟩ → ⟨Expr⟩ **suma** ⟨Expr⟩

Estructura válida

Validación de la
Estructura de entrada



Compiladores: Tarea de Análisis

Análisis SEMÁNTICO: se encarga de comprobar que se cumplan las restricciones semánticas del lenguaje tales como la necesidad de declarar las variables antes de usarlas, las reglas de tipos o la coincidencia entre los parámetros de las funciones en las definiciones y las llamadas.

Compiladores: Tarea de Síntesis

- Generación de código intermedio: es la etapa en la que se traduce la entrada a una representación independiente de la máquina y fácil de traducir a un lenguaje ensamblador
- Generación de código objeto: es la etapa en la que el código intermedio se traduce en código objeto. Es una fase totalmente dependiente de la arquitectura para la que se ha desarrollado el compilador

Compiladores: Tarea de Síntesis

- Optimización: tanto el código intermedio como el código objeto presentan traducciones ineficientes con código redundante.

Compiladores: Ventajas

- Tienden a ser más rápidos que los traducidos en tiempo de ejecución, debido a la sobrecarga del proceso de traducción.
- Se compila una vez, se ejecuta n veces.
- El compilador tiene una visión global del programa, por lo que la información de mensajes de error es mas detallada.
- Son unidades autónomas listas para ser ejecutadas.

Compiladores: Desventajas

- Necesita mucha mas memoria.
- No se puede ejecutar hasta tanto este libre de todo error.
- Dado que traduce el código fuente a un lenguaje máquina específico, los programas deben ser compilados específicamente para OS X, Windows o Linux, así como para arquitecturas de 32 o 64 bits.
- Tiempos de compilación, las grandes suites de aplicaciones pueden tardar cantidades significativas de tiempo en compilar.

Compiladores: Ejemplos

Lenguajes que utilizan Compiladores:

ADA	ORACLE J DEVELOPER
C	C++
POWER BUILDER	FORTRAN
PASCAL	DELPHI
MODULA	Eiffel

Principales Diferencias

Entre Intérpretes y Compiladores

Interprete	Compilador
El programa fuente se lee línea a línea	El programa fuente se lee totalmente
Traduce el programa cuando se ejecuta, convirtiendo el código fuente en lenguaje máquina	El proceso de compilación lo transformo en lenguaje máquina.
Cualquier programa se puede interpretar en cualquier plataforma (sistema operativo).	El archivo generado por el compilador solo funciona en la plataforma en donde se lo ha creado.
No genera un ejecutable.	Puede Generar un ejecutable..

Principales Diferencias

Entre Intérpretes y Compiladores

Interprete	Compilador
La ejecución es más lenta, ya que para cada línea del programa es necesario realizar la traducción (ej bucle)	Hablando de la velocidad de ejecución un archivo compilado es de 10 a 20 veces más rápido que un archivo interpretado.
El código fuente es necesario en cada ejecución, No funciona si no se tiene el intérprete.	El proceso de traducción se realiza una sola vez. La ejecución es independiente
Los errores sintácticos se detectan durante la ejecución, ya que traducción y ejecución se van haciendo simultáneamente.	Los errores sintácticos se detectan durante la compilación. Si el programa fuente contiene errores sintácticos, el compilador no producirá un ejecutable.

LENGUAJES INTERMEDIOS

Algunos lenguajes pertenecen a ambas categorías, es decir un programa escrito en estos lenguajes pasan por una fase de compilación intermediaria (un archivo escrito en un lenguaje ininteligible y no ejecutable) que requeriría de un interprete.

Los applets Java, pequeños programas que a menudo se cargan en páginas web, son archivos compilados que sólo pueden ejecutarse dentro de un navegador web.