Programación IV

Introducción.

Tipado, Inferencia de tipo, Lenguaje declarativo e imperativo, Lenguaje de alto nivel y bajo nivel, Lenguaje compilado e interpretado, Concepto de Paradigma de Programación. Tipos de problemas a los que atiende. Ejemplos de uso y contextos.
Los lenguajes de programación y sus paradigmas,lenguajes puros e híbridos

UNIVERSIDAD ADVENTISTA DEL PLATA

Evolución de los Lenguajes de programación

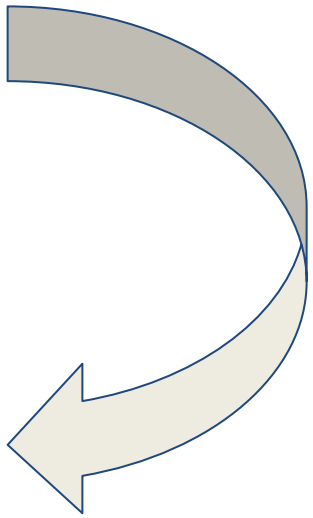
- 1940
  - "que funcione"
  - prelingual: previo a los lenguajes conocidos
- 1950
  - ensambladores
  - cálculo numérico (FORTRAN)
  - "explotación de la potencia de la máquina"
- 1960
  - COBOL, LISP, ALGOL, BASIC
  - estructuras de datos
  - recursividad
  - "aumento de la expresividad"
- 1970
  - reducción de la dependencia de la máquina: portabilidad
  - aumento de la correctitud de los programas
  - PASCAL, ALGOL 68, C
- 1980:
  - "reducción/manejo de la complejidad"
  - MODULA 2, ADA (introduce paralelismo(ejecutar con varios procesadores)-> programación concurrente (aplicaciones militares)), SMALLTALK(primer lenguaje de O.O. puro), MIRANDA.
- 1990
  - paralelo, distribuido(un programa es dividido en varias computadoras, división de tareas a través de la red. Ej.: cliente - servidor)
  - lenguajes visuales
- 2000
  - programación WEB/interacción entre varios lenguajes

Tipos

- **Tipado dinámico y tipado estático**
- **Tipado fuerte y tipado débil**

Inferencia de tipo

int a = 9;



val a = 9

Nivel

leng. Humano

ALTO

NIVEL DEL LENGUAJE DE PROGRAMACION

leng. Maquina

BAJO

Cibertronic 2010

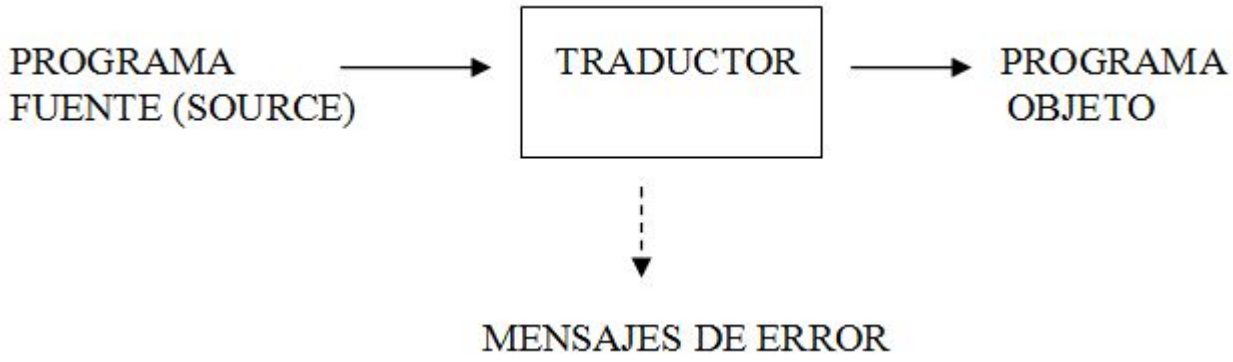
Declarativo vs Imperativo

Lenguaje compilado

Lenguaje interpretado

Traductores

Es un programa que toma como entrada un programa escrito en un lenguaje (programa fuente) y lo convierte a un programa equivalente (en cuanto a su significado) escrito en otro lenguaje (programa objeto)



Traductores

Compilador: programa que transforma un programa escrito en un lenguaje de alto nivel (por ejemplo Pascal, C, Java, Smalltalk, etc), en otro programa equivalente escrito en un lenguaje de bajo nivel (normalmente Ensamblador).

Ensamblador: convierte de bajo nivel a bajo nivel (ASSEMBLER a lenguaje máquina).

Preprocesador: convierte de alto nivel a alto nivel.

Intérprete: convierte y ejecuta instrucción por instrucción. Ej.: BASIC, PERL, SMALLTALK, PHP

Comparación entre compilador e intérprete

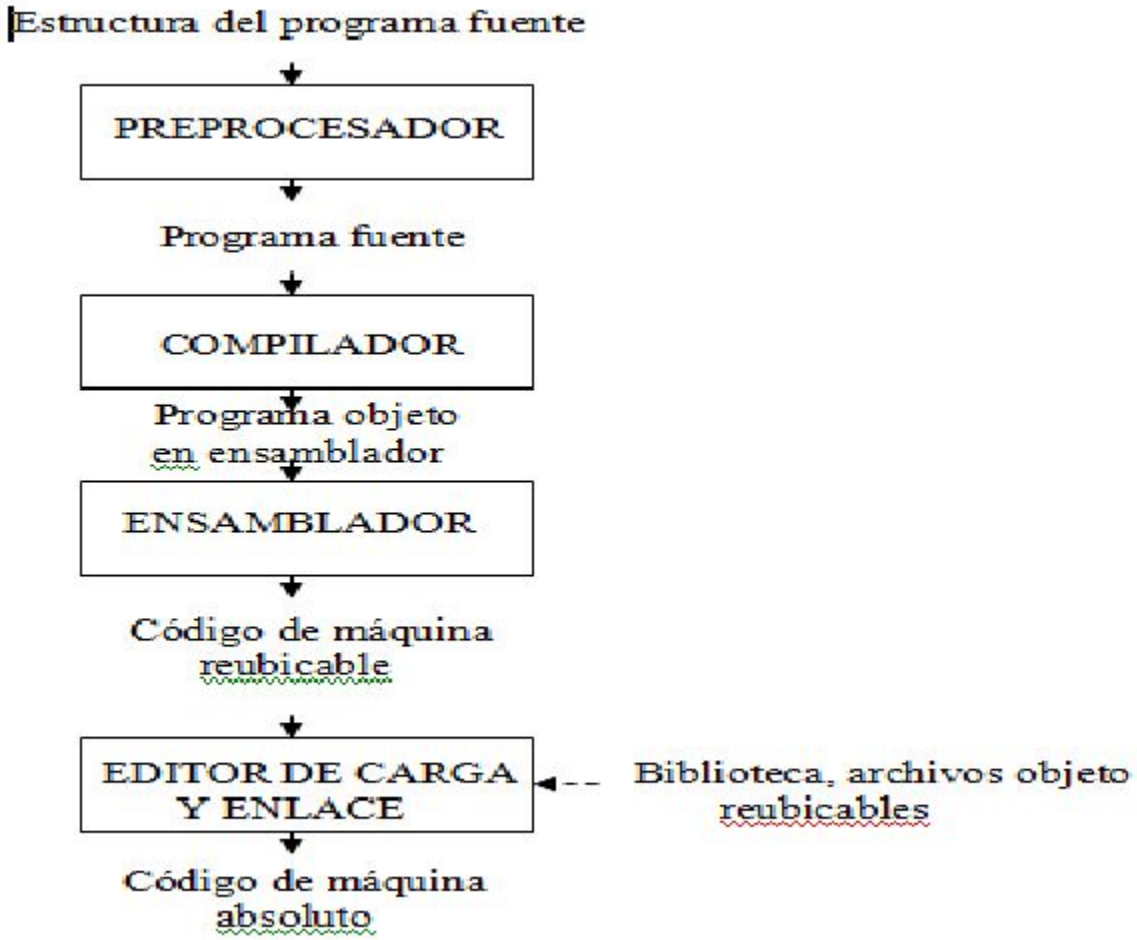
Compiladores	Intérpretes
Mayor velocidad de ejecución del programa	Facilidades en la depuración del programa
Menor requerimiento de memoria	Para lenguajes débilmente tipados
Control de algunos tipos de errores en tiempo de compilación	

Traductores

- Objetivos de la traducción
  - Conversión del programa
  - Detección de errores

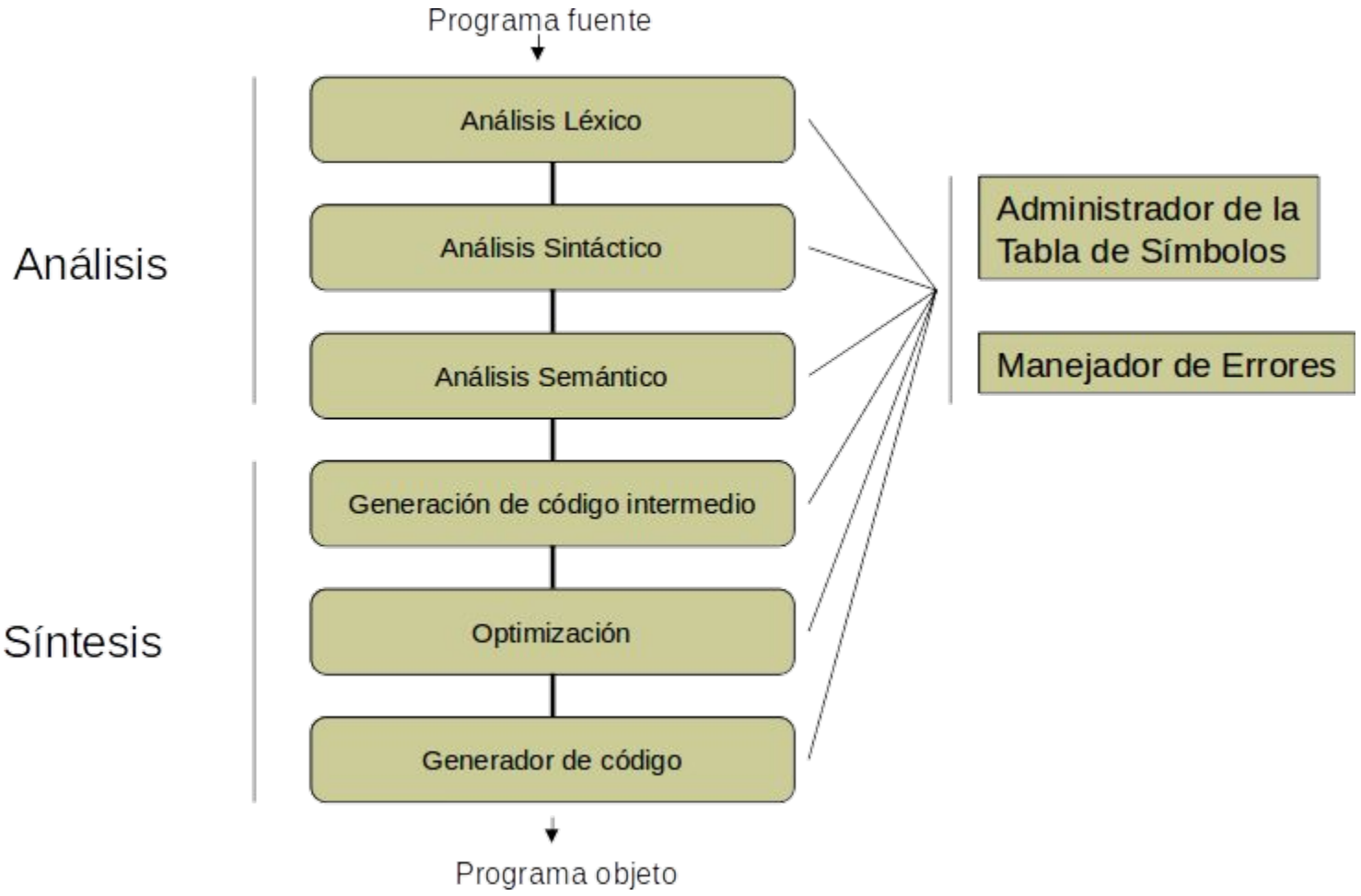
- Tipos de errores
  - Léxicos: en los componentes atómicos del lenguaje
  - Sintácticos: en la estructura del programa
  - Semánticos: en cuanto al significado del programa
    - de tipo / declaración
    - lógicos
    - conceptuales (aplicación de conceptos)
    - de comprensión del problema / requerimientos
    - Otros (dependientes de la arquitectura / S.O. / configuración

Sistema de procesamiento de un lenguaje



Proceso de compilación

Fase: división conceptual del proceso de compilación



Clasificación por Generación

1ª generación: lenguaje máquina

2ª generación: lenguaje ensamblador

3ª generación: lenguajes procedurales (imperativos)

4ª generación: lenguajes de aplicación: Ej. SQL.

Lenguajes declarativos: especifican qué se quiere hacer y no cómo se debe hacer

5ª generación: técnicas de inteligencia artificial y lenguajes de inferencia

6ª generación: redes neuronales: simulan el comportamiento de las neuronas del cerebro humano.

Paradigma

- Imperativo
- Lógico
- Funcional

¿Que es la programación funcional?

La programación funcional es un paradigma de programación declarativa basado en la utilización de funciones aritméticas que no maneja datos mutables o de estado. Enfatiza la aplicación de funciones, en contraste con el estilo de programación imperativa, que enfatiza los cambios de estado

Programación funcional

- No mantiene estados.
- Enfatiza la aplicación de funciones
- Las funciones no tienen efecto secundario
- Uso de recurrencia

Ventajas de usar un paradigma funcional

- Ausencia de efectos colaterales (transparencia referencial)
- Proceso de depuración menos problemático
- Pruebas de unidades más confiables
- Mayor facilidad para la ejecución concurrente

Lenguajes funcionales

Entre los lenguajes funcionales puros, cabe destacar a Haskell y Miranda.

Los lenguajes funcionales híbridos más conocidos son Scala, Lisp, Clojure, Scheme, Ocaml, SAP y Standard ML (estos dos últimos, descendientes del lenguaje ML). Erlang es otro lenguaje funcional de programación concurrente.

Mathematica permite la programación en múltiples estilos, pero promueve la programación funcional. R también es un lenguaje funcional dedicado a la estadística.

Recientemente Microsoft Research está trabajando en el lenguaje F# (Functional#).


UNIVERSIDAD
ADVENTISTA
DEL PLATA