Introducción
Compiladores e Interpretes


OVEJAS=0
¿ME DORMÍ?
SI
NO
OVEJAS=0
OVEJAS=OVEJAS+1
FIN
El sueño de un programador
DESMOTIVAR.COM

Compilador…

"El arte de la programación, es el arte de organizar la complejidad".

Compilador…

- Von Neumann... Introduce el concepto de programa almacenado.

- Propuso que los programas se almacenaran de forma digital en la memoria de la computadora junto con los datos.

Compilador…

La Máquina de Von Neumann

Código de Máquina

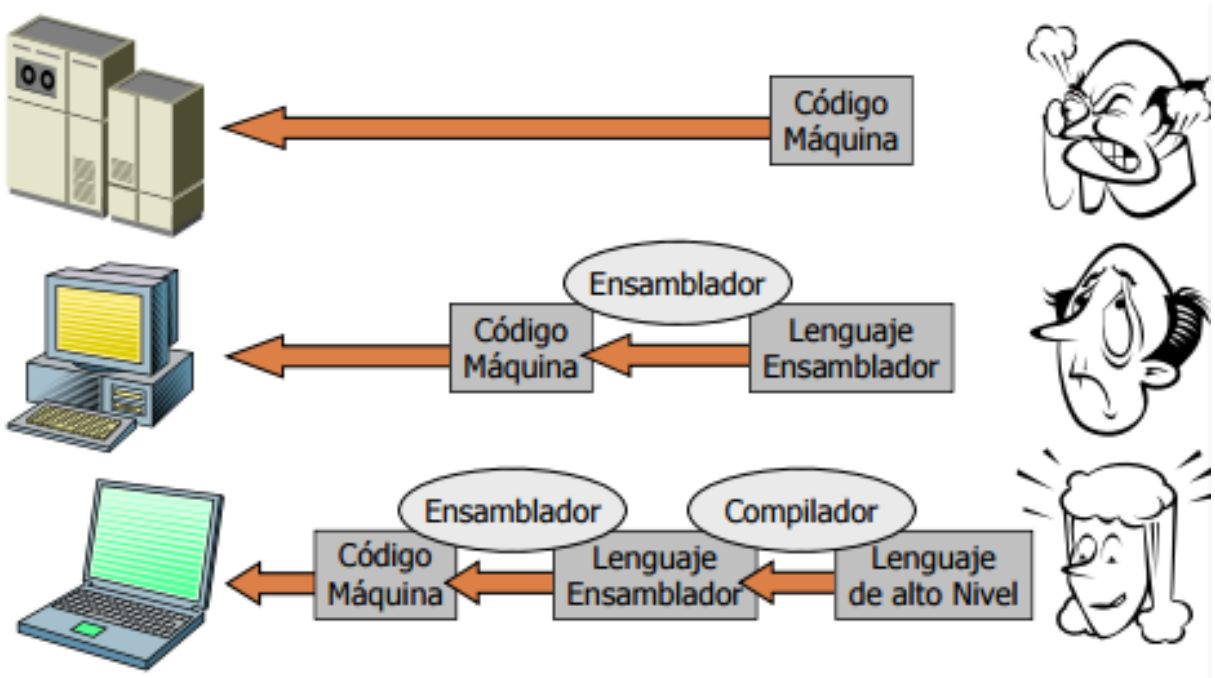
Lenguaje de máquina: 00000010101111001010
00000010101111101010
00000011001100100110

Lenguaje Ensamblador: Load I
Add J
Store K

Lenguaje alto nivel: K = I + J

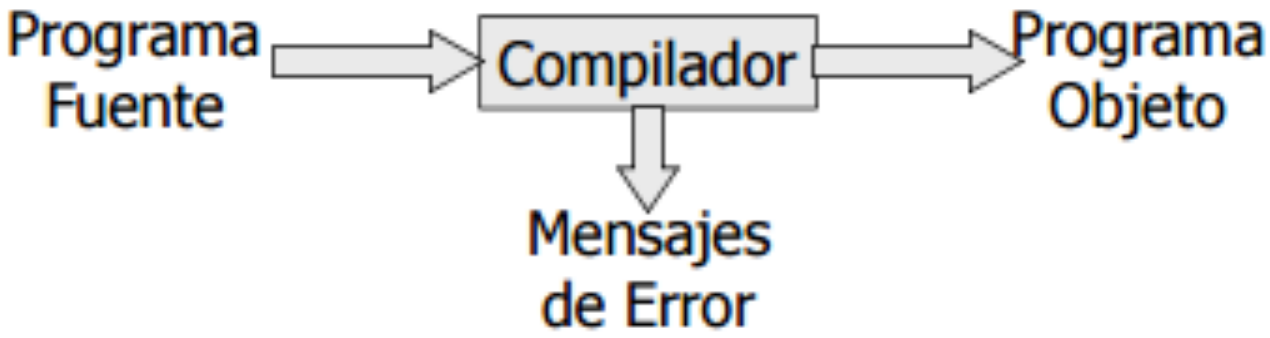
Compilador…

- Proceso de “traducción” que convierte un fuente escrito en un lenguaje de alto nivel (entendido por el usuario) a un programa código máquina (entendido por un sistema) y listo para ser ejecutado.



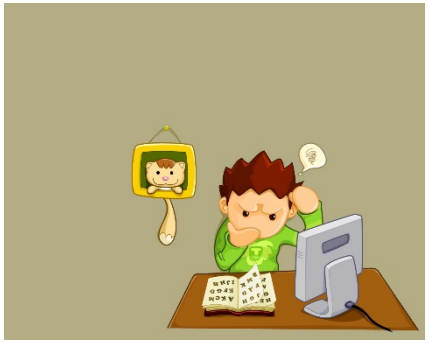
Compilador…

- Proceso de “traducción” que convierte un fuente escrito en un lenguaje de alto nivel (entendido por el usuario) a un programa código máquina (entendido por un sistema) y listo para ser ejecutado.



Por qué estudiar compiladores…

- Ser mejor programador
  - Comprender la interacción entre los lenguajes, compiladores y hardware
  - Entender de técnicas de implementación
  - Asimilar los conceptos de los depuradores de código.
  - Mejorar la intuición acerca de lo que el código hace



Por qué estudiar compiladores…

- Mezcla de teoría e Ingenieria
  - Aplicaciones dirigidas de la teoría a la práctica (parseo, scaneo, análisis estatico, etc.)
  - Algunos problemas muy difíciles
    - Localización de recursos, optimización
    - Necesidad de llegar a buenas aproximaciones / heurísticos

Por qué estudiar compiladores…

- Podría necesitar escribir un compilador algún día… no muy lejano !!!

```
main:
TRISA = 0x00          ' Configure pins as outputs

While TRUE

PORTA = 0x00          ' Turn PORTA LEDs OFF
Delay_ms(1000)        ' 1 second delay
PORTA = 0xFF          ' Turn PORTA LEDs ON
Delay_ms(1000)        ' 1 second delay

wend                  ' Endless loop
end.
```

Program written in Basic

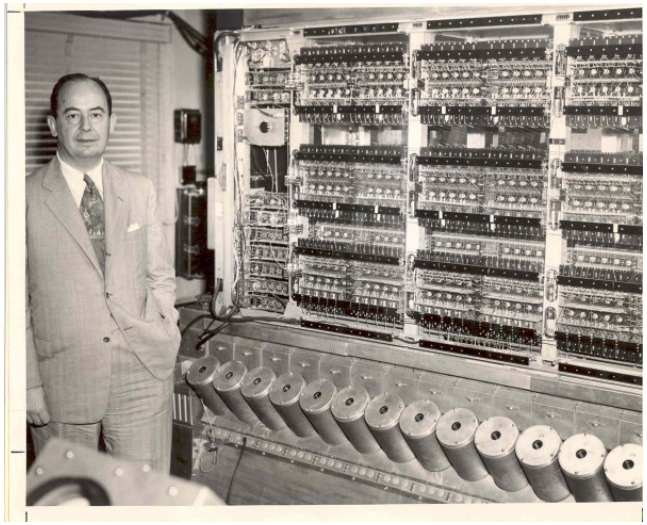
```
BSF         STATUS, 5
BCF         STATUS, 6
CLRF        TRISA
L__main2:
BCF         STATUS, 5
CLRF        PORTA
MOVLW       11
MOVWF       R11
MOVLW       38
MOVWF       R12
MOVLW       93
MOVWF       R13
L__main6:
DECFSZ      R13, 1
GOTO        L__main6
DECFSZ      R12, 1
GOTO        L__main6
DECFSZ      R11, 1
GOTO        L__main6
NOP
NOP
MOVLW       255
MOVWF       PORTA
MOVLW       11
MOVWF       R11
MOVLW       38
MOVWF       R12
MOVLW       93
MOVWF       R13
L__main7:
DECFSZ      R13, 1
GOTO        L__main7
DECFSZ      R12, 1
GOTO        L__main7
DECFSZ      R11, 1
GOTO        L__main7
NOP
NOP
GOTO        L__main2
wend                    ' Endless loop
GOTO        $+0
```

The same program compiled into assembly code. As can be seen, each Basic command is broken into several assembly instructions during the process of compiling.

Un poco de historia…

Historia…

- John Von Neumann impulsor de computadoras con programas almacenados (década de 1940)



- Se populariza la escritura de secuencias de códigos o programas que indicarán a la computadora los cálculos deseados.

- Estos programas generalmente se escribían en lenguaje máquina: códigos numéricos representando operaciones reales de la máquina.

Historia…

- En 1954 IBM desarrolla el 704
  - Sucesor del 701
  - Primera máquina comercial con éxito
  - Los costos del software excedían los costos de hardware, a pesar de que el hardware era extraordinariamente caro.
  - Investigadores en búsqueda de programación más productiva

- Los esfuerzos para mejor la productividad de la programación fue conocida como "Speedcoding" (codificación de velocidad), desarrollado en 1953 por John Backups. Ahora es conocido como interpretes.

- La ventaja es la rápidez de programar código, la desventaja era que corría entre 10 y 20 veces más lento.

Historia…

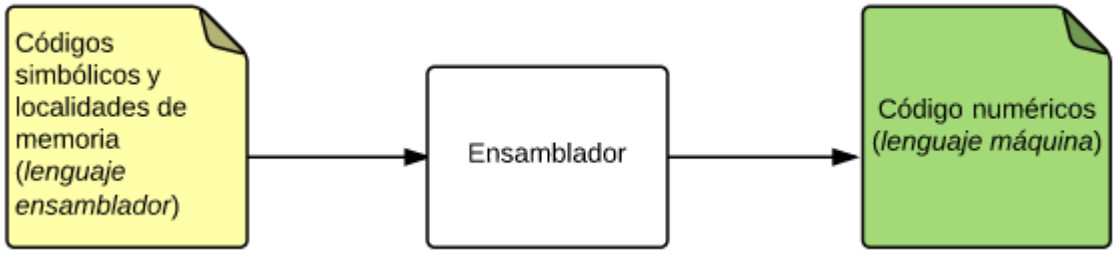
- La siguiente instrucción mueve el número 2 a la ubicación 0000 (hexadecimal) en los procesadores Intel 8x86 en las PC de IBM.

- Pronto se reemplazó por código de lenguaje ensamblador donde las instrucciones y localidades de memoria son formas simbólicas dadas.

Suponer que la localidad de memoria simbólica X es 0000

Historia…

- Ensamblador



- Mejoró la velocidad y exactitud al programar.

- Desventajas:
  - No es sencillo escribir programas
  - No es sencillo entender programas
  - Fuerte dependencia hacia la máquina escrita
  - Reescritura de código por cada máquina meta

Historia…

- El siguiente paso fue permitir una escritura más concisa similar a la notación matemática o lenguaje natural independiente de cualquier máquina en particular y que permita la traducción hacia código ejecutable.

```
1         K=1
2    6    IF (K.EQ.11) GO TO 8
3         READ,I,J
4         IF (J.GT.I) GO TO 65
5         GO TO 66
6    65   WRITE(6,6002)J,I
7    6002 FORMAT(' ',I3,' IS GREATER THAN ',I3)
8         K=K+1
9         GO TO 6
10   66   WRITE(6,6001)I,J
11   6001 FORMAT(' ',I3,' IS GREATER THAN ',I3)
12        K=K+1
13        GO TO 6
14   8    CALL EXIT
15        END
```

- El anterior código reescrito:

```
X = 2
```

- Se desarrollo el lenguaje FORTRAN (*Formulas Translated*) y su compilador.
- FORTRAN fue llevado a cabo por un equipo de IBM dirigido por John Backus entre 1954 y 1957.

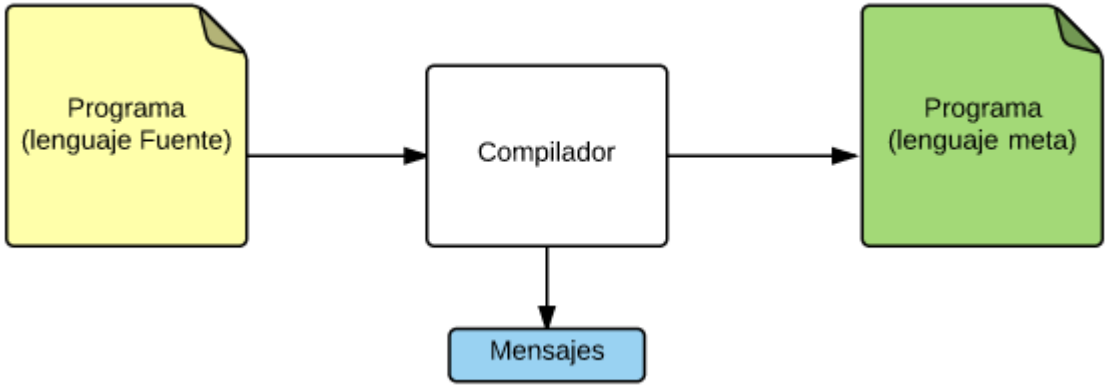
Historia…

- FORTRAN fue el primer lenguaje de alto nivel y tuvo un gran impacto en las ciencias de la computación.
- Compuesto de un enorme cuerpo de trabajo teórico, principalmente la unión de teoría más practica.
- Compiladores de automóviles modernos aún preservan el contorno de FORTRAN I

```
1           K=1
2     6     IF (K.EQ.11) GO TO 8
3           READ,I,J
4           IF (J.GT.I) GO TO 65
5           GO TO 66
6     65    WRITE(6,6002)J,I
7     6002  FORMAT(' ',I3,' IS GREATER THAN ',I3)
8           K=K+1
9           GO TO 6
10    66    WRITE(6,6001)I,J
11    6001  FORMAT(' ',I3,' IS GREATER THAN ',I3)
12          K=K+1
13          GO TO 6
14    8     CALL EXIT
15          END
```

Historia…

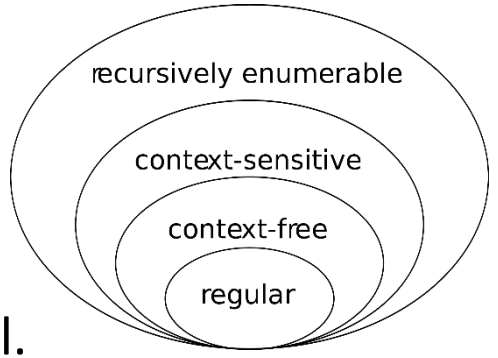
- Compiladores



Por lo general,
lenguaje de alto nivel

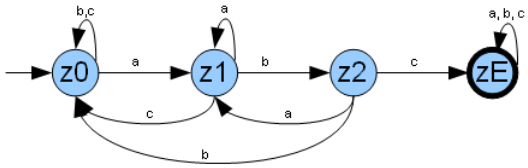
Por lo general,
código objeto (**código de máquina)** :
código escrito en las instrucciones de máquina correspondientes
a la computadora en la cual
se ejecutará

Historia…



- Noam Chomsky estudia la estructura del Lenguaje Natural.
- Facilitó la escritura de los compiladores y en cierto grado su automatización.
- A partir de esto, generó la clasificación de los lenguajes de acuerdo a la complejidad de sus gramáticas (reglas que especifican su estructura) y generación de algoritmos para reconocerlas.
- La Jerarquía de Chomsky se compone de 4 niveles: gramáticas tipo 0, tipo 1, tipo 2 y tipo 3, cada una de las cuales es una especialización de su predecesora.

Historia…



- Las gramáticas de tipo 2 (gramáticas libres de contexto) surgieron como las más útiles y estándar para lenguajes de programación.

- El problema del análisis sintáctico (determinación de algoritmos eficientes para el reconocimiento de lenguajes libres de contexto) se llevó a cabo en las décadas de los 60 y 70s.

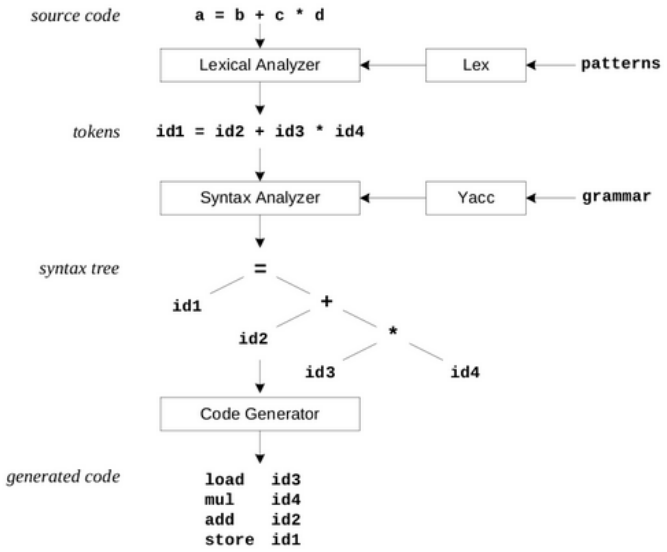
- Los autómatas finitos y las expresiones regulares (gramáticas de tipo 3 de Chomsky) se relacionan fuertemente con las gramáticas libres de contexto.

Historia…

- Se desarrollaron programas conocidos como "compiladores de compiladores" y más formalmente como "generadores de analizadores sintácticos".

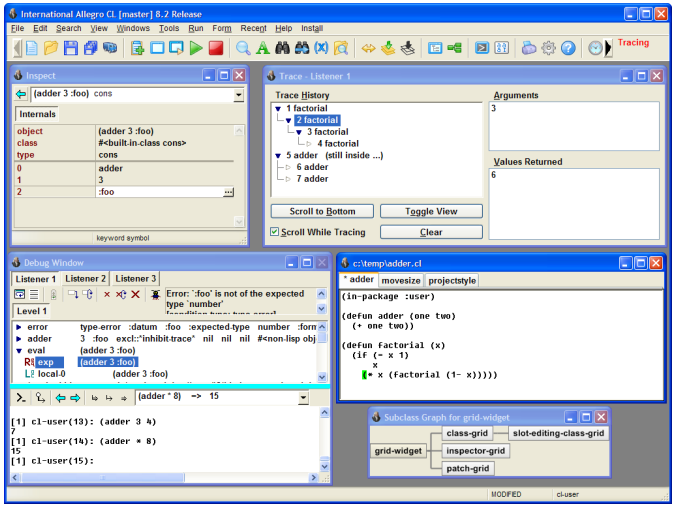
- El más conocido es el programa Yacc ("*yet another compiler-compiler*") fue escrito por Steve Johnson en 1975 para el sistema Unix.

- Similarmente, el estudio de los autómatas finitos desarrolló a los generadores de rastreadores (generador de analizadores léxicos), cuyo programa más conocido es Lex escripto por Mike Lesk para el sistema Unix.



Historia…

- Hoy en día, el diseño de compiladores han incluido aspectos tales como:
  - Algoritmos sofisticados para inferir y/o simplificar información contenida en un programa.
    - ✓ Un ejemplo de esto es el algoritmo de unificación de verificación de tipo Hindley-Milner utilizado en la compilación de lenguajes funcionales.
  - Los compiladores se han vuelto parte de ambientes de desarrollo interactivo o IDE (*interactive development environment*) basado en ventanas, que incluyen editores, depuradores, linkers y administradores de proyectos.

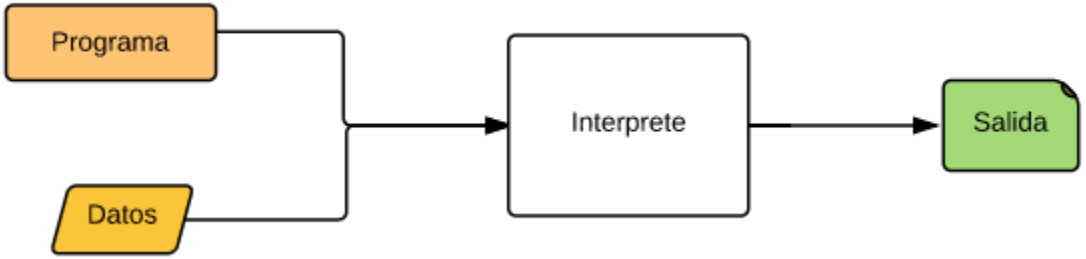


Proceso de Compilación

Introducción…

- Interprete

en línea

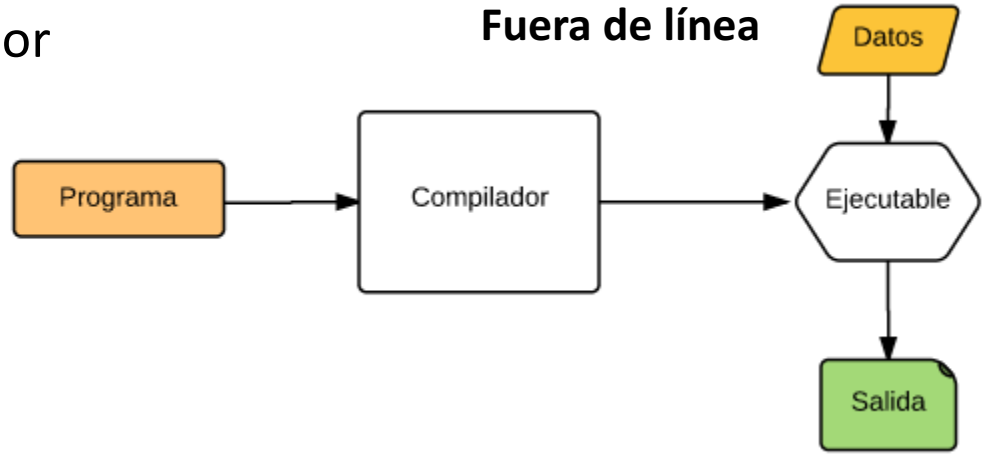


Por lo general,
lenguaje de alto nivel

Por lo general,
código objeto (**código de máquina)** :
código escrito en las instrucciones de máquina correspondientes
a la computadora en la cual
se ejecutará

Introducción…

- Compilador



Por lo general,
lenguaje de alto nivel

Por lo general,
código objeto (**código de máquina)** :
código escrito en las instrucciones de máquina correspondientes
a la computadora en la cual
se ejecutará

Introducción…

- Máquina Abstracta

- Llamado **computador abstracto**, es un modelo teórico de un sistema computador de hardware o software usado en la teoría de autómatas.

- Las máquinas abstractas con frecuencia son usadas en experimentos de pensamiento sobre computabilidad o para analizar la complejidad de algoritmos .

- El ejemplo más conocido es la máquina de Turing.

Proceso de compilación…

- Fases mayores de un compilador

1. Lexical Analysis
2. Parsing
3. Semantic Analysis
4. Optimization
5. Code Generation

Análisis Sintáctico

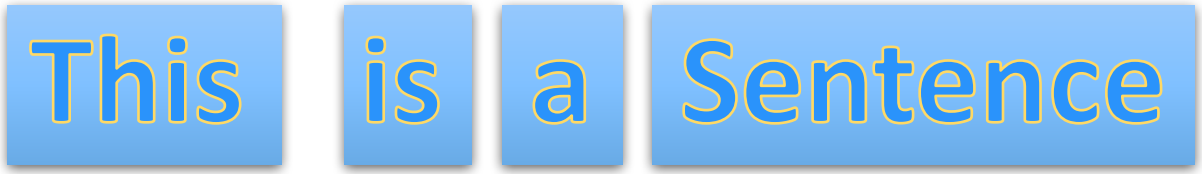
Tipos, alcance

Traducción

Análisis Léxico…

Llamado también **Scanning** o **Linear Analysis**

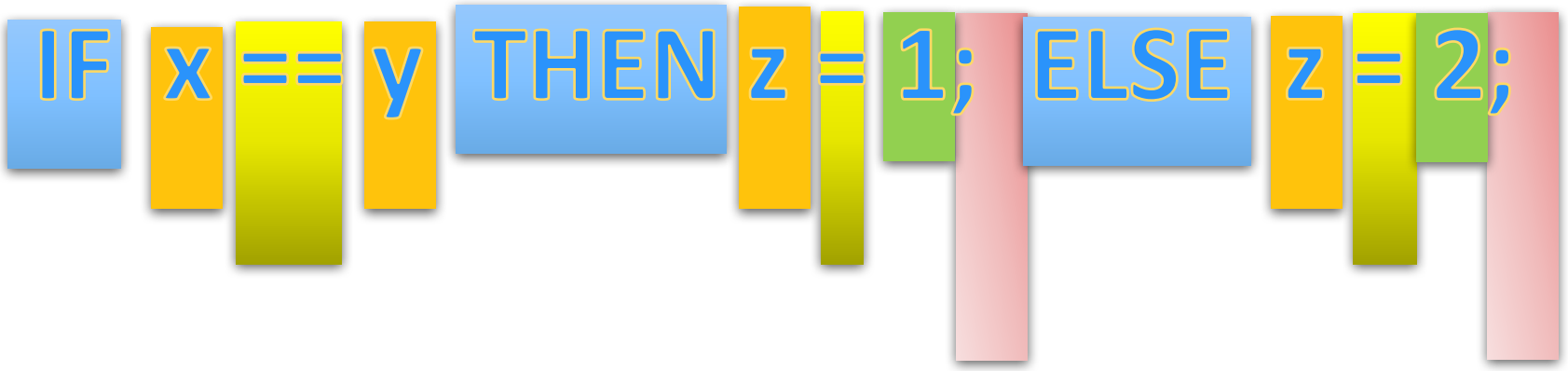
Reconocer Palabras



Análisis Léxico…

Reconocer Palabras

- Divide el texto del programa en “palabras”o “tokens”



- Distinción entre = y ==
- Conjunto de palabras que conforman un texto.

Análisis Léxico…

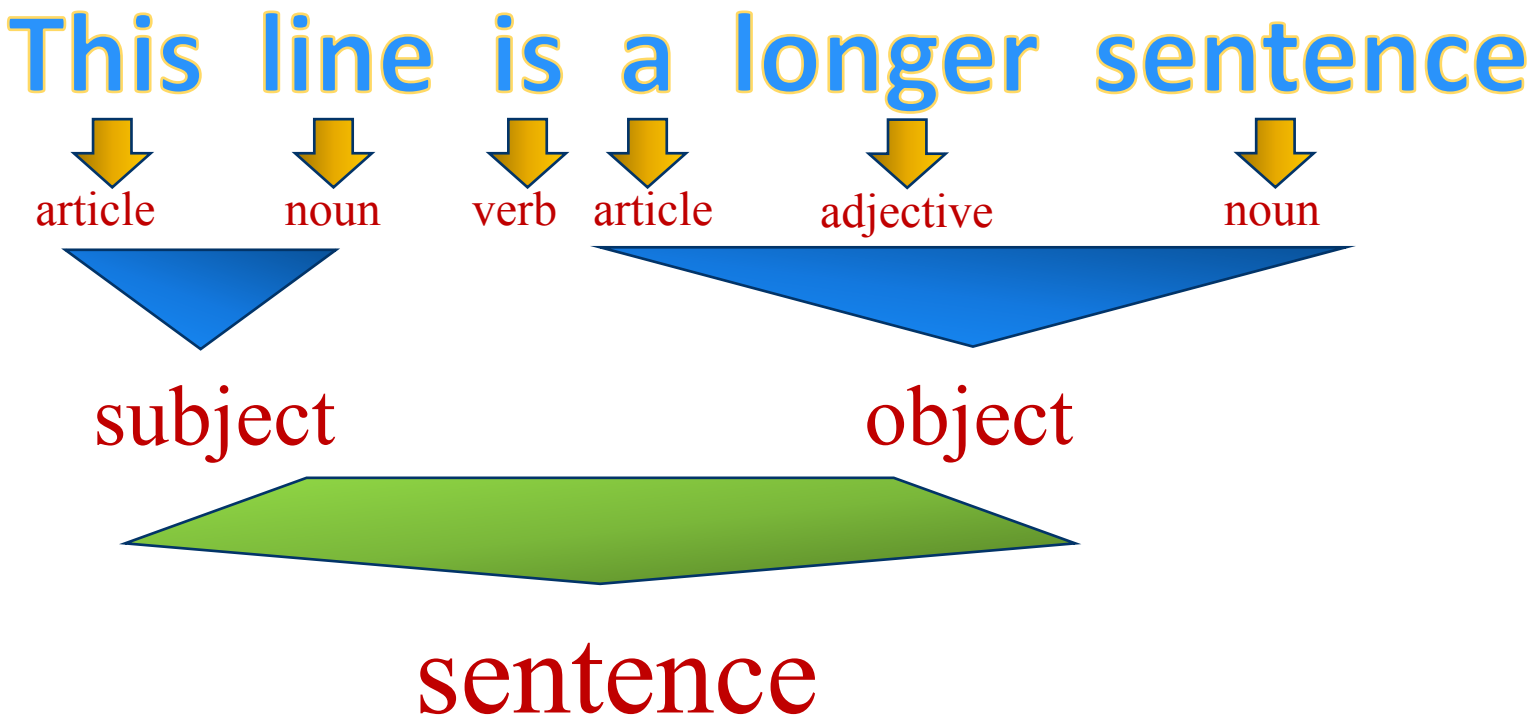
- Se leen los caracteres de un programa fuente y se agrupan en una corriente de “tokens”
- Cada token representa una secuencia lógica de caracteres
- La secuencia de caracteres formando un token es llama el “lexema” para el token.
- Una tabla de símbolos es una estructura de datos conteniendo un registro para cada identificador y sus atributos.

Parsing…

Llamado también **Syntax Analysis** o **Hierarchical Analysis**

Diagramas (árboles)

- Entender la estructura de la sentencia
- Diagramar sentencias



Parsing…

Diagramas (árboles)

IF x == y THEN z = 1; ELSE z = 2;

x == y z 1 z 2

relación asignación asignación

predicado then else

IF-THEN-ELSE

Árbol de Parsing

Análisis Semántico…

- Una vez que la estructura de la sentencia es entendida, se deben entender su "significado”
- Es una tarea complicada
- Los compiladores generalmente análisis limitados y con el objetivo de encontrar inconsistencias.

Análisis Semántico…

- Ejemplo de ambigüedad

Jack said Jerry left his assignment at home

A quién se refiere?
A Jack o a Jerry ?

```
{
 int Jack = 3;
  {
    int Jack = 4;
    cout << Jack;
  }
}
```

Cual valor de Jack se imprime?

El lenguaje posee reglas estrictas para evitar la ambigüedad

Optimización…

Da el mismo mensaje.

Similar a
~~Un poco como~~ la comedia

- La meta en la optimización de programas, es modificar el programa de tal manera que sea capaz de utilizar menos recursos.

- Automáticamente se modifican los programas de tal manera que:
  - Se ejecuten de forma más eficiente y rápida
  - Use menos memoria

Optimización…

- Se podría encontrar una regla de optimización como la siguiente:

X = Y * 0 ES LO MISMO QUE X = 0

Cierto?

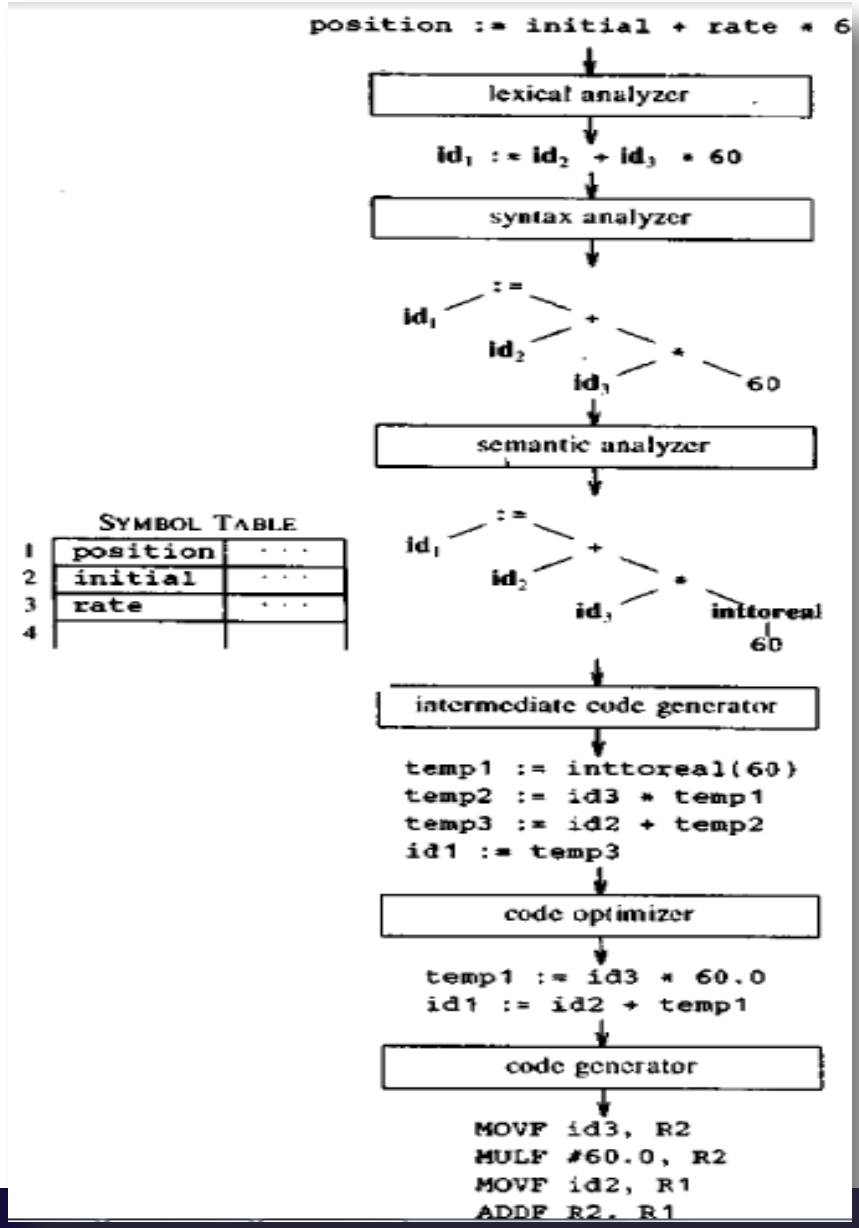
- Muchas veces lo más obvio, no necesariamente es lo correcto.
- La regla anterior es válido para Enteros (integer)
- La regla es inválida para el punto flotante, en el estándar IEEE de puntos flotantes indica.

  NAN * 0 = NAN

Generador de Código…

- Generalmente produce código en lenguaje ensamblador
- Una traducción a otro lenguaje
- Similar a la forma de traducir en un lenguaje natural.

Traducción de una sentencia…



Compiladores…

- ¿por qué existen tantos lenguajes de programación?
- ¿por qué hay nuevos lenguajes de programación?
- ¿qué es un buen lenguaje de programación?

Compiladores…

- ¿por qué existen tantos lenguajes de programación?
  - Se trata de necesidades
    - Ciencia de la computación (FORTRAN)
    - Aplicaciones de negocios (Sql)
    - Programación de sistemas (C/C++)
- ¿por qué hay nuevos lenguajes de programación?
  - Mayoría de los lenguajes son lentos al cambio
  - Fácil de iniciar un nuevo lenguaje
- ¿qué es un buen lenguaje de programación?
  - No hay una métrica universalmente aceptada para diseñar lenguajes

Introducción
Compiladores e Interpretes