Laboratorio Lenguajes Formales y de programación

Clase 2

Agenda

- Preguntas, dudas o comentarios
- Clase
  - Compilador y las partes de un compilador
  - Jerarquía de chomsky
- Explicación de la práctica.

Compilador

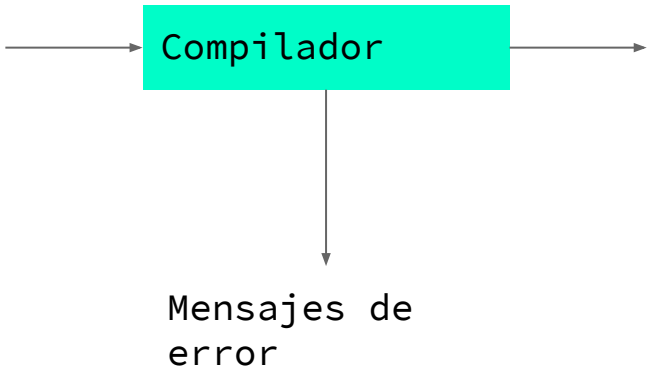
Que es un compilador

C++, C, Pascal, Cobol

Es un programa que lee un programa escrito en un lenguaje: lenguaje fuente, y lo traduce a un programa equivalente en otro lenguaje: lenguaje objeto

Programa Fuente

C++, C



Intérprete

Que es un intérprete

Ruby, Javascript, Python, etc

En lugar de producir un programa objeto como resultado de una traduccion, un inteprete realizar las operaciones que implica el programa fuente.

Compilador vsr intérprete

Compilador

- Más rápido en el ciclo de ejecución.
- No es necesario realizar la instalación de un intérprete.

Intérprete

- Más rápido en el ciclo de desarrollo.
- Multiplataforma.

Que es más rápido un compilador o intérprete?

Fases de un compilador

Analizador léxico

Lee el programa fuente de izquierda a derecha, caracter a caracter y los agrupa en componentes léxicos

Explicación

Si nosotros creamos un programa que sea capaz de realizar operaciones aritméticas simples como suma, resta, division y multiplicacion.

Componente léxicos mínimos

números enteros: 4, 10, 520

Símbolos: + - / *

Análisis sintáctico

En esta fase se agrupan los caracteres o componente léxicos de una forma jerárquica que poseen un significado colectivo.

Explicación

Primera entrada

    a = 0 (lexico y sintactico)

Segunda entrada

    a; 0 = (léxico)

Análisis semántico

Se realizan ciertas revisiones para asegurar que los componentes de un programa se ajusta de manera logica.

Explicación

Declaración y asignación de una variable.

```
int a;

a = 0;
```

a = “hola mundo” (no es válido semánticamente, pero si es válido de manera léxica y sintáctica)

Manejo de Errores

En cada fase existen errores y se debe de tratar de una manera en que la compilacion siga, lo que permitirá más detección de errores,

Error 1

```
if numero < 0:{
    print("¡Le he dicho que escriba un número positivo!")
}
print("Fin")
```

Error 2

```
if 78:
    print("¡Le he dicho que escriba un número positivo!")
print("Fin")
```

Gramáticas

Que es una gramática

Es un ente o modelo que nos permite poder especificar un lenguaje

Objetivos de una gramática

- Definir si un sentencia pertenece o no al lenguaje.
- Describir estructuralmente las sentencias del lenguaje.

Partes de una gramática

- Alfabeto o conjunto de terminales.
- Conjunto de no terminales
- Producción inicial.
- Conjunto de producciones o reglas de producción.

Ejemplo

Se desea crear un lenguaje binario (0,1) que acepte cadenas en donde el primer carácter que aparezca deberá ser 0, seguido de dos caracteres 1 y luego acepte uno o más caracteres ya sea 0 o 1.

Alfabeto o conjunto de terminales Σ

Esta parte representa el conjunto de símbolos, letras, números o palabras que aceptara nuestro lenguaje. Un terminal no puede producir más terminales.

Para el ejemplo

Σ= {0,1}

Conjunto de no terminales N

Un símbolo no terminal, es utilizado para poder representar producciones de otros símbolos terminales o no terminales.

El no terminal S puede representar al terminal 0 o a otro no terminal T

Para el ejemplo

N = {A, B, C, D}

Producción inicial, Inicio

Esta parte es la que indica cuál será el inicio de nuestro lenguaje.

Para el ejemplo

Inicio = A

Conjunto de producciones o reglas de producción

Representa los pasos por los cuales debera de atravesar una cadena para ser valida en nuestro lenguaje.

```
A -> 0 B
B -> 1 C
C -> 1 D
D -> 0 D
D -> 1 D
D -> ε(epsilon)
```

Dudas hasta el momento?

Jerarquía de chomsky

Gramáticas tipo 0

Máquina de turing

Conocidas como gramáticas no restringidas o recursivamente enumerables

Restricciones

$$\propto \rightarrow \beta$$

$\propto$ = puede contener más de un símbolo, siempre que exista al menos un no Terminal.
β = No existen restricciones en β, puede ser una cadena de terminales y no terminales.

Gramáticas tipo 1

Autómata linealmente acotado

Conocidas como gramáticas dependientes o sensibles del contexto.

Restricciones

$$\propto A\beta \rightarrow \propto \gamma\beta$$

A = es un no terminal

El lado izquierdo puede contener más de un símbolo, siempre que exista al menos un no terminal.

$\propto\gamma\beta$ = son cadenas de terminales y no terminales, $\propto\beta$ pueden ser vacíos, pero γ tiene que ser distinto de vacío.

La longitud del lado izq <= longitud del lado der.

Dependientes del contexto

Se denomina de esa forma dado que:

$$\propto A\beta \rightarrow \propto \gamma\beta$$

El símbolo no terminal **A** puede ser sustituido por γ sí está acompañada de ∝ por la izquierda y de β por la derecha.

Gramáticas tipo 2

Autómata descendente (Pila)

Conocidas como gramáticas independientes del contexto. Este tipo de gramáticas son las utilizadas en la fase analisis sintactico.

Restricciones

$$A \rightarrow \beta$$

A = es un no terminal y no acepta mas simbolos.

β = es una cadena de terminales y no terminales.

Independientes del contexto

A -> β

Se denomina independientes del contexto dado que **A** puede ser sustituido por β independientemente de las cadenas por las que esté acompañada.

Gramáticas tipo 3

Automata Finito

Conocidas como gramáticas regulares. Los lenguajes regulares se utilizan para definir la estructura léxica de los lenguajes de programación (Analisis Lexico).

Restricciones

```
A -> aB
A -> a
```

Un único no terminal en la parte izquierda.

Una parte derecha compuesta por un único terminal que puede estar o no acompañado de un no terminal.

Importante

Todo lenguaje de tipo 3 es de tipo 2, todo lenguajes de tipo 2 es de tipo 1, y todo lenguaje de tipo 1 es de tipo 0.