UNIVERSIDAD TECNOLÓGICA DE SANTIAGO (UTESA)

ÁREA DE ARQUITECTURA E INGENIERÍA
CARRERA DE INFORMÁTICA

PROGRAMACIÓN DE VIDEO JUEGOS

COMPILADORES
TAREA SEMANA 7

Presentado a:

Ing. Iván Mendoza

Presentado por:

Luis B. Espinal 1-18-0635

Santiago de los Caballeros Rep.Dom, noviembre del 2024

Documentación Extendida del Compilador

Implementación un compilador simple en Python, utilizando la biblioteca PLY (Python Lex-Yacc) para procesar un lenguaje básico que admite:

1. Declaración y asignación de variables.
2. Operaciones aritméticas básicas: suma, resta, multiplicación y división.
3. Uso de paréntesis para cambiar la precedencia de operaciones.

El compilador abarca las fases esenciales del proceso de compilación:

- Análisis Léxico.
- Análisis Sintáctico.
- Análisis Semántico.
- Tabla de Símbolos.
- Generación de Código Intermedio.

2. Requisitos del Sistema

Antes de ejecutar este proyecto, asegúrate de cumplir con los siguientes requisitos:

1. Python 3.6+
2. Biblioteca PLY

Características del Lenguaje Soportado

El lenguaje diseñado para este compilador admite:

1. Variables
   - Los nombres de variables deben comenzar con una letra o guion bajo (_) y pueden contener números.
   - Ejemplo: x = 1;

2. Operaciones Aritméticas
   - Soporta operadores básicos: +, -, *, /.
   - Ejemplo: y = x + 5;

3. Precedencia de Operadores

- Los operadores tienen precedencia estándar: multiplicación y división tienen mayor precedencia que suma y resta.
- Los paréntesis pueden cambiar el orden de evaluación.
- Ejemplo: z = (x + y) * 5;

4. Declaración Obligatoria

Las variables deben declararse antes de ser usadas.

5. Finalización de Declaraciones

Cada declaración debe terminar con un punto y coma (;).

6. Estructura del Compilador

El compilador tiene las siguientes etapas:

- Analizador Léxico

  Escanea el código fuente y lo divide en tokens, que son las unidades más pequeñas del lenguaje.

- Analizador Sintáctico

  Interpreta los tokens generados por el léxico y construye un Árbol de Sintaxis Abstracta (AST).

- Analizador Semántico

  Verifica la validez de las operaciones y las declaraciones de variables.

- Tabla de Símbolos

  Almacena información sobre las variables y sus valores.

- Generador de Código Intermedio:

  Traduce el AST en una representación más sencilla.

Ejemplo de Uso

Código Fuente:
x = 1;
y = x + 2;
z = (x + y) * 5;

Salida Generada:
AST: ('assign', 'x', 1)
Código intermedio: 1
Tabla de símbolos: {'x': 1}

AST: ('assign', 'y', ('+', 1, 3))
Código intermedio: (1 + 3)
Tabla de símbolos: {'x': 1, 'y':4}

AST: ('assign', 'z', ('*', ('+', 5, 8), 2))
Código intermedio: ((5 + 8) * 2)
Tabla de símbolos: {'x': 5, 'y': 8, 'z': 26}

Manejo de Errores

- Errores Léxicos:

Entrada: x = 5 @;
Salida: Caracter no válido: @

- 2. Errores Sintácticos:

Entrada: x = ;
Salida: Error de sintaxis.

- Errores Semánticos:

Entrada: y = z + 3;
Salida: Error: Variable no declarada: z