Universidad Interamericana de Panamá

Facultad de ingeniería y de sistemas

Informe Final del Compilador

Carrera

Ingeniería en sistemas computacionales

Integrantes

Suehellen Méndez 4-753-1861

Josué De León 8-876-2357

Asignatura

Compiladores

Facilitador

Leonardo Esqueda

Fecha

22 de agosto de 2018

1. **Objetivos**

* El objetivo de este trabajo es la construcción de un compilador.
* Programaremos dos analizadores: léxico y sintáctico.
* Nuestro analizador debe decirnos que significa una palabra

1. **Lenguaje**

En esta sección se presentan aspectos del lenguaje de programación Python.

1. **Arquitectura**

Para la arquitectura en compilador se presentan dos fases, de análisis y de síntesis.

En la fase de análisis lee el código fuente luego divide los tokens y hace el análisis léxico y sintáctico posteriormente, luego de hacer el análisis para a la segunda fase y ejecuta el programa, tabla de símbolos este puede tener varias pasadas.

**¿Qué es una fase de un compilador?**

Es una etapa distinguible del proceso de compilación, que toma como entrada la salida de la etapa anterior. La fase procesa esta información y el resultado es entonces pasado a la siguiente fase. Una pasada puede tener más de una fase.

1. **Diagrama T**

Mensaje de error

Lenguaje Objetivo

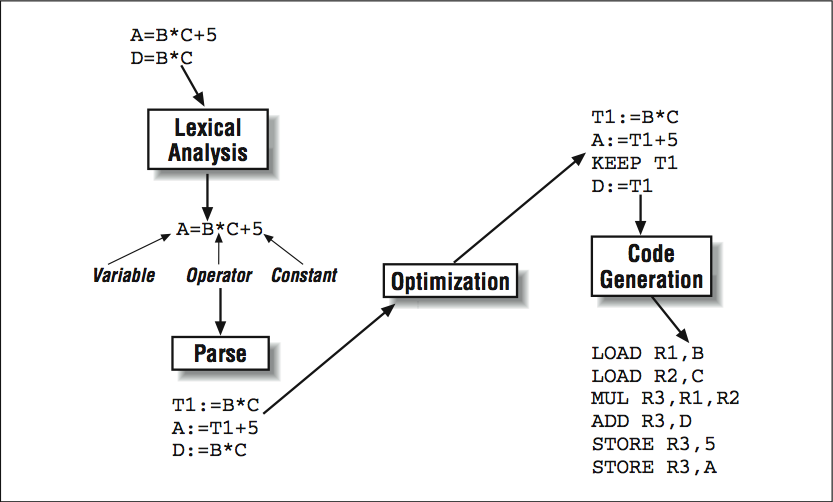
Python

Compilador

Código Fuente

Python

1. **Diagrama de flujo**

****

1. **Descripción de la utilidad**

Contamos con dos analizadores: Analizador léxico que viene siendo la primera fase para el compilador este recibe como entrada el código fuente de otro programa y produce una salida compuesta de tokens o símbolos. Mientras que el analizador sintáctico analiza una cadena de símbolos de acuerdo a las reglas de una gramática formal.

Nuestro analizador se encarga de ir analizando variable por variable hasta llegar al final e imprime que es la variable.

**Librería Yacc:** YACC significa "otro compilador de compiladores más" (del inglés Yet Another Compiler-Compiler), lo que refleja la popularidad de los generadores de analizadores sintácticos al principio de los años setenta, cuando S.C. Johnson creó la primera versión de Yacc. Este generador se encuentra disponible como una orden del sistema Unix y se ha utilizado para facilitar la implantación de cientos de compiladores**.**

**Librearía:** LEX es una herramienta utilizada en la especificación de analizadores léxicos para varios lenguajes. Esta herramienta se denomina compilador Lex (PCLEX para el entorno DOS), y la especificación de su entrada, lenguaje Lex. El estudio de una herramienta existente permitirá mostrar cómo, utilizando expresiones regulares, se puede combinar la especificación de patrones con acciones, por ejemplo, haciendo entradas en una tabla de símbolos, cuya ejecución se pueda pedir a un analizador léxico. Se pueden utilizar las especificaciones tipo Lex aunque no se disponga de un compilador Lex; las especificaciones se pueden transcribir manualmente a un programa operativo empleando técnicas de diagramas de transiciones.

1. **Descripción de la tabla de símbolos**

Aquí se definió el orden de prioridad de cada token y lo que hace cada token, dentro de los cuales tenemos:

('right' , 'ASSIGN'), #Operador de asignación

('right' , 'UPDATE'), #Operador de actualización

('left' , 'NE'),

('left' , 'LT' , 'LTE' , 'GT', 'GTE'), #Espacio y tabulación

('left' , 'PLUS' , 'MINUS'), #Suma y resta

('left' ,'TIMES' , 'DIVIDE'), #Multiplicación y división

('right' , 'ODD'), #Sirve para ver si una expresión es impar

('left' , 'LPARENT' , 'RPARENT'),

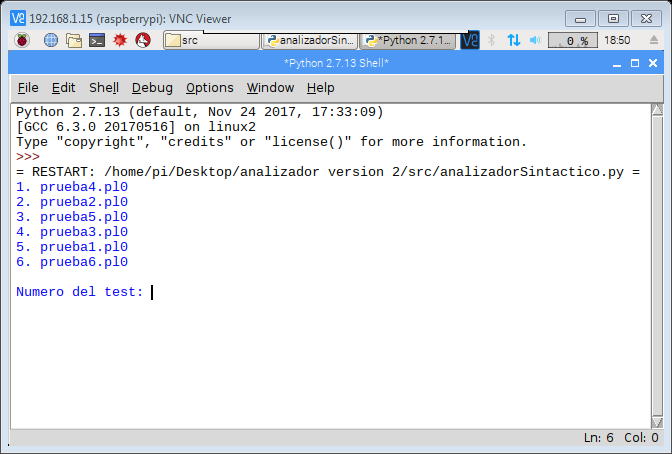
1. **Como maneja los errores**

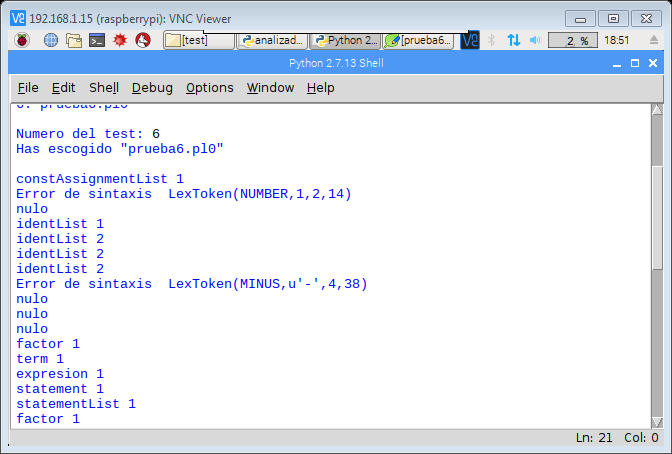
Se declara la variable para que indique cuando hay error y que especifique en que línea, que tipo de error se está presentando en el analizador.

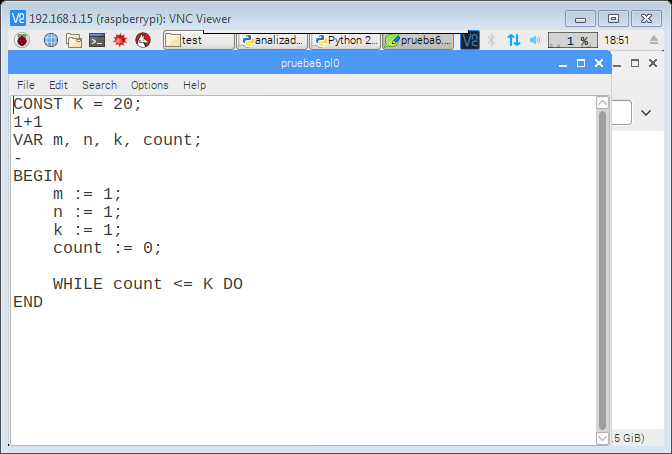
1. **Código de programación en GITHUB**

<https://github.com/JOSUEXLION/Compiladores/tree/master/analizadores/Compilador_PL0-master/analizador%20version%202>

**Ejemplos de uso**

****

****

****