Universidad Rafael Landívar

Facultad de ingeniería

Ingeniería en informática y sistemas, sección 01

Catedrático: Ing. Moisés Alonso

**PROYECTO NO. 1**

**“ANALIZADOR LÉXICO”**

Ivania Alejandra Velásquez Morales 1045718

Guatemala, 4 de marzo del 2020

**INTRODUCCIÓN**

El proyecto que se presentará el miércoles 4 de marzo del 2020 consiste en realizar la primera fase que es un analizador léxico por medio de un programa. En este caso para el proyecto que se presentará a continuación está hecho en el lenguaje de alto nivel llamado PYTHON.

Un analizador léxico lee los caracteres de entrada y como salida tiene una secuencia de componentes léxicos, esta es la primera fase de un compilador. Este tiene como función en la interfaz eliminar del programa los comentarios que se realizan, espacios en blanco en forma de caracteres de espacio en blanco, saltos de línea y también relaciona los mensajes de error, tal que diga que hay un carácter invalido, o una sintaxis inválida en el código que existe dentro del archivo que se está probando en el programa del analizador léxico. Como se dijo antes se trabajó por PYTHON, en los links bibliográficos se tendrá todo lo que se descargó para poder trabajar en esto, cabe recalcar que se trabajó con la versión de PYTHON 3.8.2, también se debe decir que no se trabajó en el Shell de PYTHON sino que se descargó una aplicación de terceros para tener mayor comodidad a la hora de trabajar, que se llama SUBLIME TEXT 3 y también estará el link de descarga en la bibliografía, también se debe de decir que lastimosamente no se pude realizar un autómata o un árbol para realizar el proyecto entonces se tuvo que usar una librería de PYTHON para poder realizar parte de la fase.

**OBJETIVOS**

GENERAL

Comprender la función del analizador léxico y sintáctico de un compilador a través de un programa.

ESPECÍFICO

En este caso como el proyecto cuenta en tres fases, en esta se quiere comprender la fase del analizador léxico.

**ANÁLISIS DEL PROYECTO**

ENTRADAS

Para este programa se tienen los tokens y palabras reservadas, también entradas de espacios vacíos, saltos de línea. En este caso se hace carga de un archivo de cualquier tipo para que el programa lo analice línea por línea.

PROCESOS

Al principio del programa tiene tokens, palabras reservadas declaradas para que no las vaya a tomar como un error cuando corra, las palabras reservadas para que las tome como tokens se concatena con estos y las reservadas lo manda a una lista y en el recorrido toma en cuenta sus valores y así ya los toma como tokens. Luego de esto, cada token se fue declarando, como una expresión regular para que el programa lo fuera reconociendo, para los símbolos de ER se fueron tomando con \ para que no hubiera ningún problema. Luego de las declaraciones se programaron funciones para que reconocieran función del token como el token ID, el que quiere decir que en la ER puede comenzar cualquier letra de la a-z ya sea minúscula o mayúscula y que eso puede venir 0 o muchas veces, y en eso también pueden venir números del 0-9. Se tiene una condición para que reconozca las mayúsculas del token y se tienen su función de return para que imprima el ID. Se tiene una función que en el programa también reconozca un comentario como una ER, un “.” que lo reconoce como un carácter comodín lo único que no reconoce es un salto de línea, esta no devuelve ningún token ya que la función no es muy requerida en esta fase, también se tiene un método que reconoce cualquier número decimal, al valor del token lo envía como un número entero y así lo regresa en el return. Se tiene una función error la cual reconoce si hay algún token inválido, un token que no se reconoce en el programa, quiere decir que no se reconoce en las funciones y también imprime el carácter que nos está dando el problema, este reconoce los tokens desde la posición 0. Se tiene un método que también reconoce los espacios, ya que se pide que no importa cuantos espacios en blanco tenga y no se reconozca como un error. Se tiene la ruta de archivo para que el programa los vaya solo leyendo con ayuda de la función que tiene PYTHON, que se llama CODECS, está va acompañada de una ‘r’ que es default y significa “read mode” seguido de un utf-8 que nos ayuda en la lectura de bytes del archivo. Luego de eso se cierra con un close para que no deje ningún archivo abierto. Tenemos una cadena para que lea todo lo que se encuentre en el texto con ayuda de un input que nos ayuda a convertirlo a strings. De último se tiene un ciclo que nos permite imprimir cada token hasta que ya no se encuentren más el programa y deja de analizar tokens.

SALIDAS

Análisis por línea y por columna de la oración.

**CONCLUSIONES**

Esta primera fase de proyecto es muy importante para un compilador, ya que en esta fase se puede ver como el programa va leyendo línea por línea para ver cuáles son los id o tokens los cuales se tienen escritos en los archivos.

**BIBLIOGRAFÍA**

Función analizador léxico (2019). [En red]. Recopilado el 26/02/2020. Disponible en: <http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro32/21_funcin_del_analizador_lxico.html>

Conceptos básicos analizador léxico (2019). [En red]. Recopilado el 26/02/2020. Disponible en: <http://www.cartagena99.com/recursos/alumnos/apuntes/ININF2_M4_U2_T1.pdf>

Expresiones regulares (2019). [En red]. Recopilado el 26/02/2020. Disponible en: <https://docs.microsoft.com/en-us/dotnet/standard/base-types/regular-expression-language-quick-reference>

CODECS- PYTHON (2019). [En red]. Recopilado en 1/03/2020. Disponible en:

<https://docs.python.org/3/library/codecs.html>

<https://docs.python.org/2/library/codecs.html>

<https://docs.python.org/3/howto/unicode.html>

<https://docs.python.org/3/tutorial/inputoutput.html>

PYTHON DESCARGA:

<https://www.python.org/downloads/>

SUBLIME TEXT:

<https://www.sublimetext.com/3>

PLY- PYTHON:

<https://www.dabeaz.com/ply/>