

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERIA
ESCUELA DE CIENCIAS Y SISTEMAS
LABORATORIO DE LENGUAJES FORMALES DE PROGRAMACION

PROYECTO #2
MANUAL DE USUARIO

DENNIS MAURICIO CORADO MUÑOZ
CARNET: 202010406
CUI: 3032329780108
GUATEMALA, GUATEMALA, 28/10/2021

Objetivos y Alcances

Objetivos generales

Aplicar los conocimientos de clase y laboratorio para poder realizar la creación de un analizador Sintáctico por medio de la teoría de autómatas y así dar solución al problema planteado.

Objetivos específicos

- Analizar un archivo con una estructura especificada y poder obtener información y validar su estructura
- Realizar un analizador sintáctico que determine errores en el archivo.
- Brindar información detallada de la estructura del archivo por medio de un reporte que contiene la lista de los tokens analizados, errores léxicos y errores sintácticos.

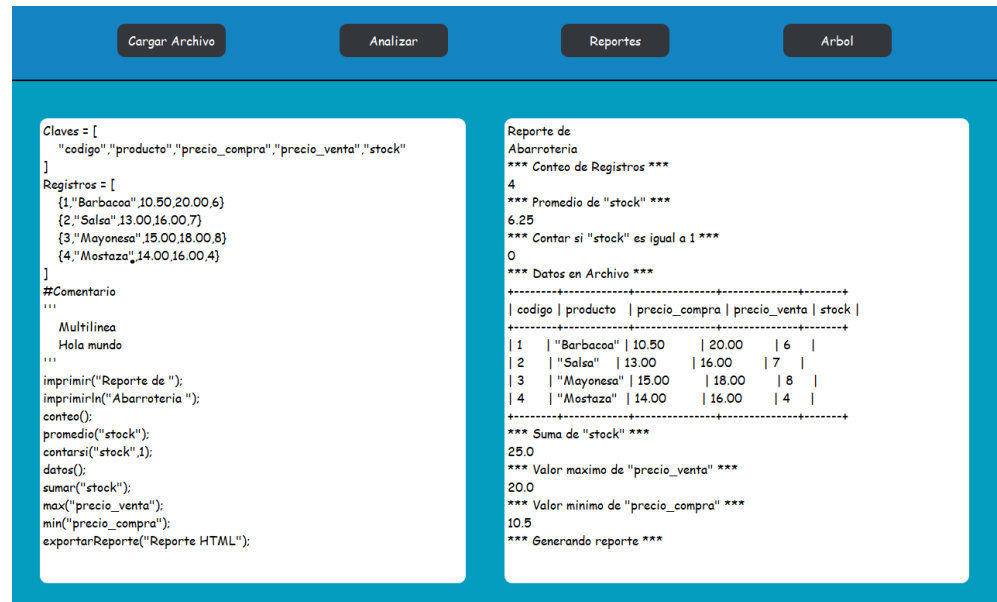
Requisitos del Sistema

- **Procesador:** Intel Pentium o cualquier superior a este
- **RAM:** 512 MB de RAM o cualquier cantidad superior a esta medida.
- **Espacio en Disco:** 750 MB libres en disco mínimo.
- **Sistema operativo:** Windows, macOS o Linux
- Tener instalado el software de Python 3.9 o superior

Interfaz Gráfica

Pantalla Inicial

La aplicación cuenta únicamente con una pantalla principal, esta pantalla contiene 4 botones y 2 cajas de texto. El funcionamiento de esta aplicación se divide en la siguiente forma, el uso de los botones debe ser secuencial debido a la información y procesos que ejecuta cada botón. El botón de “cargar archivo” como su nombre lo indica, sirve para seleccionar un archivo de nuestro equipo que



cumpla con la estructura para poder ser analizado, una vez seleccionado el texto de dicho archivo será desplegado en el cuadro blanco a la izquierda, este cuadro blanco es una caja de texto que nos permite ver la información del archivo que se selecciono así como a su vez poder editarlo por si se deben o se quieren hacer cambios a la estructura de este .

El siguiente botón, de nombre “Analizar”, de igual forma su nombre es bastante explicito respecto a lo que realiza, realiza un análisis a la información del archivo seleccionado o también al texto que se coloque dentro de la caja de texto, cualquier cambio que se haga a la estructura del archivo desde dicha caja, al presionar el botón analizar leerá el cambio realizado, de que forma trabaja este botón?, Al ser presionado primero procede a verificar que la estructura del archivo, símbolos, cadenas, palabras reservadas , etc. estén bien escritas y que no halla nada que no este definido dentro de la simbología base, posteriormente realiza un análisis sintáctico que verifica la forma en la que vienen los datos para poder ser leídos, verifica que las palabras reservadas así como las instrucciones cumplan con la estructura base para poder ser procesadas, esto es lo que realiza el botón análisis, si dentro del archivo o texto escrito existen instrucciones se proceden a ser mostradas en la caja de texto de la derecha, esta caja de texto tiene deshabilitado el poder cambiar la información de salida debido a que son datos obtenidos de la información previamente ingresada. Algunas instrucciones que se pueden darle al programa son, imprimir, imprimirln, conteo, contarsi, datos, promedio, entre otras. Algunas de estas instrucciones solo funcionan cuando se tienen los contenidos como claves y registros debido a que su resultado depende de estos campos.

Botón “reportes”, este botón se encarga de mostrar los reportes necesarios de la aplicación, entre estos reportes encontramos el reporte de tokens, que devuelve todos los tokens reconocidos por la aplicación, esta información se presenta por medio de una tabla en formato HTML, reporte de errores, este reporte contiene la información de los errores léxicos y sintácticos determinados dentro de la aplicación para poder identificar donde hubieron fallos y poder ser corregidos, de igual forma la información de este reporte se representa por medio de una tabla en un archivo HTML.

Botón “árbol”, este botón se encarga de realizar el árbol de derivación por medio de la herramienta 'graphviz', este árbol contiene todos los tokens, y lexemas reconocidos al momento de ingresar una instrucción o un archivo al programa, cabe recalcar que este árbol se genera únicamente si el archivo no cuenta con errores sintácticos, por lo tanto es importante verificar la información en el archivo ingresado y visualizar los reportes para ver si esta información no contiene errores de ningún tipo.

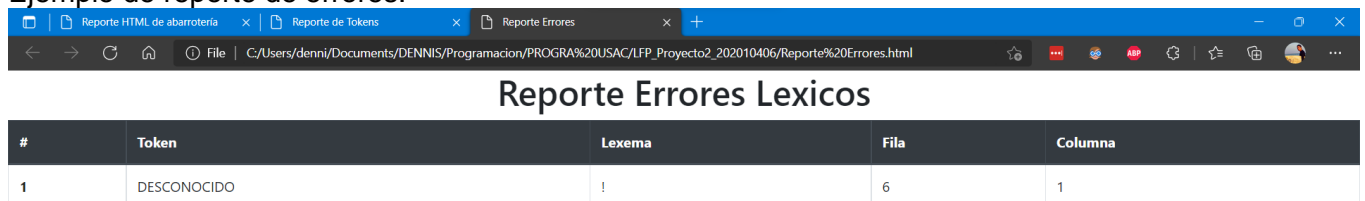
Posteriormente una vez entendido el funcionamiento de cada parte de la interfaz de este programa, en las siguientes imágenes se muestran capturas de la apariencia de los reportes y la forma como estos estarán estructurados.

Ejemplo de reporte de la instrucción 'exportarReporte();':



Reporte de Registros				
codigo	producto	precio_compra	precio_venta	stock
1	"Barbacoa"	10.50	20.00	6
2	"Salsa"	13.00	16.00	7
3	"Mayonesa"	15.00	18.00	8
4	"Mostaza"	14.00	16.00	4
5	"Frijol"	10.50	20.00	1
6	"Azucar"	13.00	16.00	3
7	"Carton de Huevos"	15.00	18.00	2
8	"Sal en libra"	14.00	16.00	4
9	"Consome"	10.50	20.00	6
10	"Arroz"	13.00	16.00	0
11	"Cafe en sobre"	15.00	18.00	0
12	"1/2 litro de agua pura"	14.00	16.00	4

Ejemplo de reporte de errores:



Reporte Errores Lexicos				
#	Token	Lexema	Fila	Columna
1	DESCONOCIDO	!	6	1

Reporte de Errores Sintacticos					
#	Token	Lexema	Fila	Columna	Esperado
1					

Ejemplo de reporte de Tokens:

Reporte de Tokens Validos				
#	Token	Lexema	Fila	Columna
1	COMENTARIO	"" Seccion de Datos""	5	4
2	IMPRIMIR	imprimir	7	9
3	PARENTESIS_IZQUIERDO	(7	10
4	CADENA	"/***** *****/	7	31
5	PARENTESIS_DERECHO)	7	32
6	PUNTO_COMA	;	7	33
7	IMPRIMIR	imprimir	8	9
8	PARENTESIS_IZQUIERDO	(8	10
9	CADENA	"Claves"	8	18
10	PARENTESIS_DERECHO)	8	19
11	PUNTO_COMA	;	8	20
12	IMPRIMIR	imprimir	9	9
13	PARENTESIS_IZQUIERDO	(9	10