Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Lenguajes Formales de Programación
Segundo semestre de 2022

Sección	Catedrático	Tutor Académico
A+	Ing. Otto Amílcar Rodríguez Acosta	Carlos Esteban Godínez Delgado
A-	Ing. Vivian Damaris Campos González	Mario Josué Solís Solórsano
B+	Ing. David Estuardo Morales Ajcot	Diego Andrés Obín Rosales
B-	Inga. Zulma Karina Aguirre Ordoñez	Pablo Daniel Rivas Marroquin

PROYECTO NO. 1

1. DESCRIPCIÓN GENERAL

1.1 OBJETIVO GENERAL

Que el estudiante cree una herramienta la cual sea capaz de reconocer un lenguaje, dado por medio de un analizador léxico el cual cumple con las reglas establecidas, manejando la lectura y escritura de archivos para el manejo de la información.

1.2 OBJETIVOS ESPECÍFICOS

- Implementar por medio de estados un analizador léxico.
- Utilizar funciones de manejo de cadenas de caracteres en lenguaje Python.
- Programar un Scanner para el análisis léxico.
- Construir un scanner basándose en un autómata finito determinístico.
- Crear una herramienta para interactuar de forma visual con el usuario

1.3 DESCRIPCIÓN

Se solicita la lectura de código fuente, creando un programa el cual sea capaz de identificar un lenguaje dado, identificando los errores léxicos y ejecutando las instrucciones correspondientes.

Se listarán una serie de instrucciones las cuales deben de ser ejecutadas, cumpliendo con el formato asignado, generándolo en un archivo HTML.

De igual manera, se deberán de manejar los errores encontrados por medio de un archivo HTML.

2. CARACTERÍSTICAS DE LA SOLUCIÓN

Para responder a las necesidades que se le plantean, se ha pensado en el desarrollo de una aplicación en lenguaje Python que permita reconocer las distintas instrucciones, y la ejecución de las mismas. Con el objetivo que se implemente el análisis léxico correspondiente.

Se deben de mostrar de manera funcional y agradable al usuario resumen de errores detectados, así como el resultado de las operaciones realizadas en cada una de las funciones que se describen más adelante.

2.1 DISEÑO DE LA INTERFAZ

En la aplicación se deberán demostrar como mínimo los siguientes menús, tomando en cuenta que deberá de ser totalmente gráfica.

Archivo	**Ayuda**
Abrir	Manual de Usuario
Guardar	Manual Técnico
Guardar Como	Temas de Ayuda
Analizar	
Errores	
Salir	

- Menú
    - **Abrir**: Permite abrir un archivo para poder seguir editándolo
    - **Guardar**: Permite guardar el archivo que está siendo editado.
    - **Analizar**: Analiza el texto y mostrará los elementos reconocidos.
    - **Errores**: Muestra los errores del último archivo compilado.
    - **Salir**: Con esta opción se cerrara la aplicación.

- Ayuda
- Ayuda
    - **Manual de Usuario**: Se deberá mostrar el manual de usuario realizado por el estudiante para guiar a la persona en la utilización de la aplicación. Este debe de estar en formato PDF.
    - **Manual Técnico**: Se deberá mostrar el manual técnico realizado por el estudiante para entender como se ha realizado la aplicación. En donde se evidencie el diseño y análisis previo para darle solución al proyecto.
        - Este debe estar en formato PDF.
    - **Temas de Ayuda**: Permite mostrar información del estudiante que ha creado la aplicación.

2.1.2 DEFINICIÓN DE OPERACIONES VÁLIDAS

Función	Operación
SUMA	Suma de 2 o más números u operaciones.
RESTA	Resta de 2 o más números u operaciones.
MULTIPLICACION	Multiplicación de 2 o más números u operaciones.
DIVISION	División entre números u operaciones.

POTENCIA	Potencia N de un número u operación.
RAIZ	Raíz N de un número u operación.
INVERSO	Inverso de un número u operación.
SENO	Función trigonométrica seno de un número u operación.
COSENO	Función trigonométrica coseno de un número u operación.
TANGENTE	Función trigonométrica tangente de un número u operación.
MOD	Residuo entre números u operaciones.

2.2 ESTRUCTURA DEL ARCHIVO DE ENTRADA

```
<Tipo>
<Operacion= SUMA>
      <Numero> 4.5 </Numero>
      <Numero> 5.32 </Numero>
</Operacion>
<Operacion= RESTA>
      <Numero> 84 </Numero>
      <Numero> 33.7 </Numero>
</Operacion>
<Operacion= MULTIPLICACION>
      <Numero> 5 </Numero>
      <Numero> 7 </Numero>
</Operacion>
<Operacion= DIVISION>
      <Numero> 45 </Numero>
      <Numero> 9 </Numero>
</Operacion>
<Operacion= SUMA>
      <Numero> 5.4 <Numero>
      <Operacion= MULTIPLICACION>
            <Numero> 7.8 </Numero>
```

```
            <Numero> 4.3  </Numero>
      </Operacion>
</Operacion>
</Tipo>

<Texto>
Realizar las operaciones básicas de suma, resta, multiplicación y
división así como operaciones complejas.
</Texto>



<Funcion = ESCRIBIR>
      <Titulo> Operaciones  </Titulo>
      <Descripcion> [TEXTO] </Descripcion>
      <Contenido>  [TIPO] </Contenido>
</Funcion>

<Estilo>
      <Titulo Color=AZUL Tamanio=12/>
      <Descripcion Color=VERDE Tamanio=3/>
      <Contenido Color=GRIS Tamanio=3>
</Estilo>
```

2.3 ESTRUCTURA DEL ARCHIVO DE SALIDA

```
1  Generacion Archivo HTML
2  Operación suma 1:
3  4.5+5.32=9.82
4
5  Operación suma 2:
6  84+33.7=50.3
7
8  Operación multiplicacion 1:
9  5*7 =35
10
11 Operación division 1:
12 45/9=5
13
14 Operación compleja 1:
15 5.4+(7.8*4.3)=38.94
```

2.4 ESTRUCTURA DEL ARCHIVO DE ERRORES

Si existieran errores léxicos, debe de generar una tabla de errores en formato **HTML**, con nombre **ERRORES_<<no_carne>>**, cumpliendo con el siguiente formato:

No.	Lexema	Tipo	Columna	Fila
1	¿	Error	5	6
2	¡	Error	8	19
3	~	Error	17	4

2.5 FUNCIONALIDAD DE LA APLICACIÓN

El código será cargado en un área de texto, se podrá modificar, guardar el archivo con el mismo nombre, o bien guardar el archivo con diferente nombre, y al momento de analizar se deberán mostrar los respectivos reportes en un archivo HTML, deberá mostrar los errores encontrados en un archivo HTML, indicando que provocó el error.

Así mismo deberá mostrar en un archivo HTML el resultado de todas las etiquetas ejecutadas, así como los texto y funcionalidad descrita anteriormente. **RESULTADOS_<<no_carne>>**

2.6 DOCUMENTACIÓN

- Manual técnico debe explicar de forma clara la lógica de su programa, el AFD del analizador léxico, los tokens y sus respectivos patrones. El analizador léxico programado debe coincidir con el AFD
- Manual de usuario
- Archivos de prueba

3 NOTAS IMPORTANTES

- El proyecto se deberá realizar en forma individual.
- El proyecto deberá implementarse en lenguaje Python, en caso contrario este no será calificado.

- Se valorará la calidad de la información proporcionada por la aplicación cuando se produzcan errores, así como la presentación de la interfaz gráfica y amigabilidad de la aplicación.
- Copias, totales o parciales de proyectos tendrán una nota de 0 puntos y las sanciones por parte de la Escuela de Sistemas.
- Sistema Operativo Libre

3.1 INDICACIONES DE ENTREGA

- Debe incluir ejecutable
- Debe incluir también Código Fuente
- La entrega se realizará en la plataforma UEDI o la indicada por el auxiliar de curso en su momento, si UEDI no estuviese disponible. Todos los archivos solicitados deberán ser entregados en un archivo zip identificado de la siguiente forma: **[LFP]Proyecto1_carnet.zip,** el estudiante es responsable de verificar que dentro del archivo zip se encuentren todos los archivos necesarios para su calificación.
- Documentación acorde a la solicitada por el auxiliar
- No será permitida la modificación de archivos de entrada durante la calificación.
- La calificación deberá ser en línea y se estará grabando la reunión para tener un respaldo de la forma en que se procedió.
- La calificación de la práctica será personal y durará como máximo 30 minutos, en un horario que posteriormente será establecido.
- **Copia parcial o total del proyecto tendrá una nota de 0 puntos, y se notificará a la Escuela de Sistemas para que se apliquen las sanciones correspondientes.**
- **En el caso de no cumplir con alguna de las indicaciones antes mencionadas, NO se calificará la practica; por lo cual, se tendrá una nota de cero puntos.**

Fecha de entrega: 22 de septiembre de 2022 antes de las 23:59 horas.