Facultad de Ingeniería

Escuela de Ciencias y Sistemas

Lenguajes Formales y de Programación

Sección B+

Manual Técnico

Interfaz Gráfica e Intérprete

Creado por: *Selim Idair Ergon Castillo*

ID: *201801300*

Contenido

1. Introducción

El siguiente manual va a dirigido a la persona o entidad que requiera el uso de la aplicación para su correcta y eficiente ejecución en el sistema operativo que posea (En este caso, el SO de Microsoft, Windows). Dicho manual presenta y describe aspectos técnicos informáticos para el manejo de archivos, y el uso de métodos y funciones para la resolución de problemas.

2. Objetivo

Otorgar una guía sencilla al usuario para la correcta ejecución del programa, tomando en cuenta los requisitos que requiere el sistema y los requerimientos que espera el mismo usuario del programa.

3. Requisitos del sistema

- **Hardware:**
    1. *CPU, monitor, teclado, mouse.*
    2. *Memoria RAM 8GB.*
    3. *Procesador 2.1GHz*

- **Software:**
    1. *Sistema operativo de 64 bits (Windows 11 en adelante)*
    2. *Python 3.10.0b3, MSC v.1926 64 bits para Windows 11*
    3. *Librería Tkinter, Librería FPDF*
    4. *Visual Studio Code, versión 1.75 para Windows 11*

4. Descarga e instalación de Python

Para la ejecución del programa, se necesita el uso del software de programación Python. Este software proporcionara la librería Tkinter para la representación de una interfaz gráfica durante la ejecución del programa.

i. ***Descarga e instalación:***

1. *Buscar en un navegador web la versión de Python que requiere la PC.*
2. *Seleccionar y entrar a la página de Python.*
3. *Seleccionar la versión requerida.*
4. *Seleccionar la opción de Descargar.*
5. *Una vez descargado, se procede a buscar el archivo descargado .EXE para su instalación.*

5. Descarga e instalación de Visual Studio Code

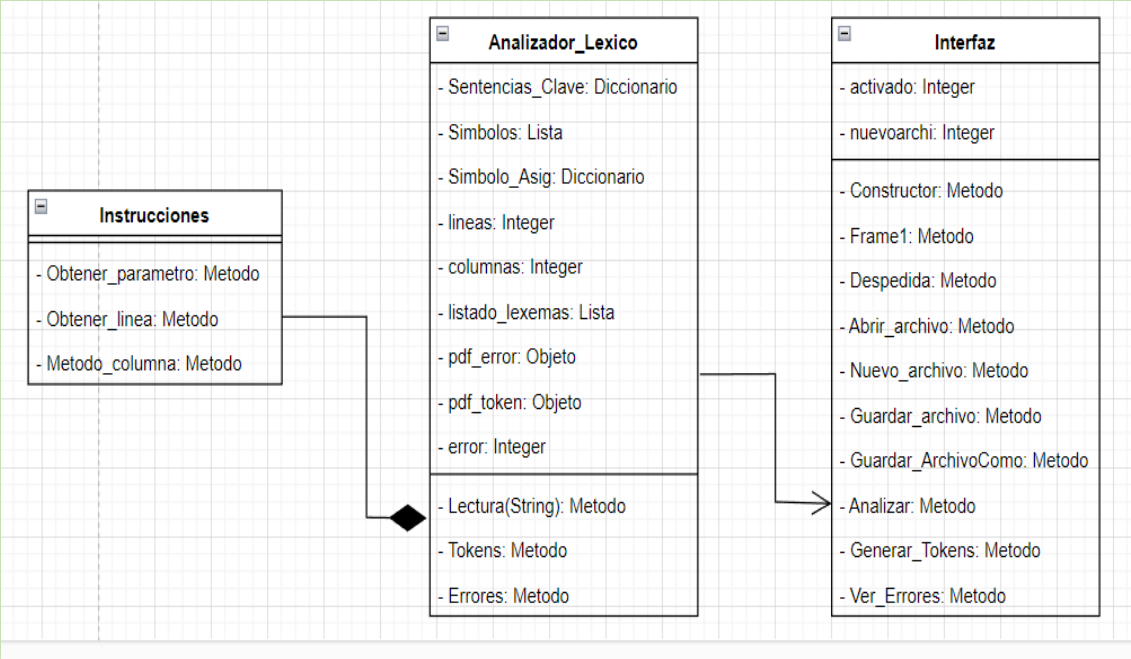
Para la ejecución de Python y la librería Tkinter, se necesita utilizar el software de grupos de lenguajes de programación, Visual Studio Code. Visualizará y hará posible la manipulación del lenguaje Python y hacer la simulación de una base de datos.

ii. ***Descarga e instalación del Visual Studio Code:***

1. *Buscar en un navegador web: Visual Studio Code.*
2. *Seleccionar y entrar a la página oficial.*
3. *Seleccionar la versión que requiere la PC.*
4. *Instalar la versión.*

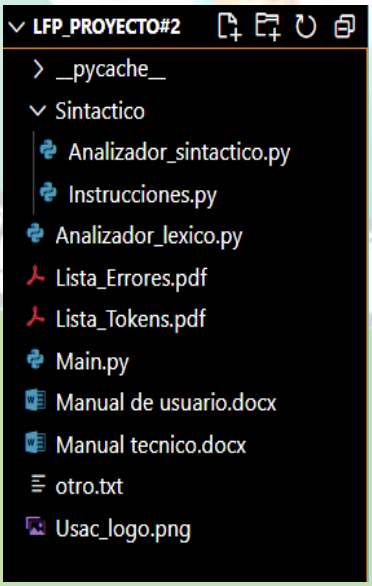
6. Diagrama de relaciones

A continuación, se muestra el diagrama relacional de los elementos usados en el programa:



7. Estructura raíz

El proyecto Python tiene la siguiente estructura de directorios:



i. ***Carpeta Sintáctico:*** *Contiene los archivos que se encargan de los análisis sintácticos.*

   a. ***Analizador_sintactico:*** *Archivo que manipula los caracteres analizados léxicamente.*

   b. ***Instrucciones:*** *Archivo abstracto encargado de dar las instrucciones sintácticas.*

ii. ***Analizador léxico:*** *Archivo que manipula los caracteres de un archivo.*

iii. ***Main:*** *Archivo principal que contiene la interfaz gráfica interactiva.*

8. Clases:

a. ***Class interfaz:*** *Clase que se encarga de crear la interfaz gráfica.*

b. ***Class instruccion:*** *Clase abstracta hija que se encarga de reconocer dígitos, retornando su ubicación en filas y columnas.*

c. ***Class Impresion:*** *Clase que imprime una cadena.*

d. ***Class Identificador:*** *Clase que determina los identificadores según el comando a realizar.*

e. **Class Asignacion:** *Clase que asigna una sentencia o comando.*

9. Paradigmas

Los paradigmas utilizados en este programa estuvieron basados:

i. ***Paradigma imperativo:*** *Se escribieron líneas de código, influenciada por el flujo de codificación top-down. Cada instrucción dada realizaba una función específica.*

ii. ***Paradigma Programación orientada a objetos:*** *Algunas clases heredan atributos de otras clases para la lectura, almacenaje y manipulación de datos durante la ejecución del programa.*

10. Constructores

Métodos que instancian el objeto de una clase perteneciente a la misma. Cada clase tiene su propio constructor. Cada constructor se define con la sentencia clave: ***def __ini__(self).***

11. Métodos y funciones

A continuación, se describen los métodos y funciones utilizados en el programa:

- ***Clase Interfaz:***
  1. *Def Frame1()*
  2. *Def Despedida()*
  3. *Def Abrir_archivo()*
  4. *Def Nuevo_archivo()*
  5. *Def Guardar_archivo()*
  6. *Def Guardar_ArchivoComo()*
  7. *Def Analizar()*
  8. *Def Generar_tokens()*
  9. *Def Ver_errores()*

- ***Clase Analizador_lexico:***
  1. *Def Lectura()*
  2. *Def Tokens()*
  3. *Def Errores()*

- ***Clase Instruccion:***
  1. *Def Operar_parametro()*
  2. *Def Obtener_linea()*
  3. *Def Obtener_columna()*

12. Tabla de Tokens

No.	*Token*	*Lexema*	*Patrón*	*Definición*
1	*Palabra reservada*	*CrearBD*	*Palabra reservada del sistema*	*Crea una base de datos.*
2	*Palabra reservada*	*EliminarBD*	*Palabra reservada del sistema*	*Elimina una base de datos.*
3	*Palabra reservada*	*CrearColeccion*	*Palabra reservada del sistema*	*Crea una colección.*
4	*Palabra reservada*	*EliminarColeccion*	*Palabra reservada del sistema*	*Elimina una colección.*
5	*Palabra reservada*	*InsertarUnico*	*Palabra reservada del sistema*	*Inserta un registro simple.*
6	*Palabra reservada*	*ActualizarUnico*	*Palabra reservada del sistema*	*Actualiza un registro simple.*
7	*Palabra reservada*	*EliminarUnico*	*Palabra reservada del sistema*	*Elimina un registro simple.*
8	*Palabra reservada*	*BuscarTodo*	*Palabra reservada del sistema*	*Busca todos los registros.*
9	*Palabra reservada*	*BuscarUnico*	*Palabra reservada del sistema*	*Busca solamente un registro simple.*
10	*Palabra reservada*	*Nueva*	*Palabra reservada del sistema*	*Crea la función dependiendo de la palabra reservada.*
11	*Palabra reservada*	*$set*	*Palabra reservada del sistema*	*Asigna un nuevo valor a un parámetro.*
12	*Simbología comentario corto*	*---*	*---, #, /*	*Comentario de una línea.*
13	*Simbología abrir comentario largo*	*/**	*/*, //*, ///**	*Comentario de muchas líneas.*
14	*Simbología cerrar comentario largo*	**/*	**/*, *//*, *///*	*Comentario de muchas líneas.*
15	*Símbolo separador agrupador*	*(*	*(*	*(*

16	*Símbolo separador agrupador*	*)*	*)*	*)*
17	*Símbolo Asignación*	*=*	*=*	*=*
18	*Símbolo sentencia final*	*;*	*;*	*;*
19	*Símbolo identificador*	*:*	*:*	*:*
20	*Símbolo parámetro*	*,*	*,*	*,*
21	*Símbolo cadena*	*“*	*“*	*“*
22	*Símbolo separador agrupador*	*{*	*{*	*{*
23	*Símbolo separador agrupador*	*}*	*}*	*}*

13. Autómata

a. *Inicia la lectura al recibir los siguientes caracteres: ‘\n’, ‘\t’, ‘ ’, ‘\s’. Se pasa a un estado siguiente.*

b. *Se recibe otro carácter. Si se recibe un carácter no estipulado en el léxico del programa (~,`, |, ¿, /, <, >, ¡, @, #, $, %, ^, &, *, -, +), se toma como error. Pasa a un estado de espera.*

c. *Sigue la lectura. Se recibe un carácter [a-zA-Z], procede a generar un lexema. Se tiene un estado de espera.*

d. *Si el lexema es una palabra reservada, se tiene un estado de aceptación.*

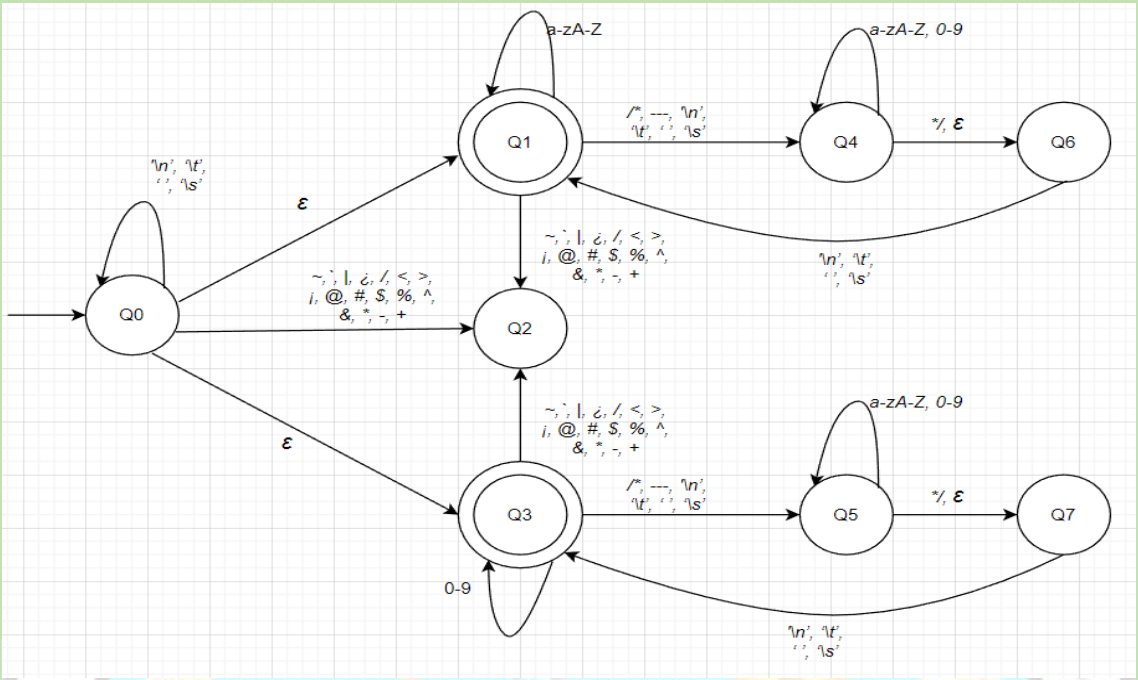
e. *Si el lexema no es una palabra reservada, se le clasifica como identificador. Se tiene un estado de aceptación.*

f. *Sigue la lectura. Se determina que un carácter es un digito. Se procede a generar un lexema. Se tiene un estado de aceptación.*

g. *Se recibe otro carácter. Si se recibe un carácter no estipulado en el léxico del programa (~,`, |, ¿, /, <, >, ¡, @, #, $, %, ^, &, *, -, +), se toma como error. Pasa a un estado de espera.*

h. *Si se reconoce un "/*", "*/" o "---", se determinan que son comentarios. Pasa a un estado de espera.*

i. *Después de se haya procedido a la lectura total de lexemas, se procede a leer el ultimo carácter para la finalización del autómata.*



Componentes: *A = (Q, Σ, δ, Q0, F)*

1. *Q es su conjunto de estados: Q0, Q1, Q2, Q3, Q4, Q5, Q6, Q7*
2. *Σ son los símbolos de entrada: Caracteres alfanuméricos.*
    a. *δ es la función de transición.*
3. *Q0 es el estado inicial*
4. *F es el conjunto de estados de aceptación: Q1, Q3.*

14. Gramática

a. ***Init → Instrucciones***

b. ***Instrucciones → CREAR_BASE;***

| ELIMINAR_BASE;

| CREAR_COLL;

| ELIMINAR_COLL;

| INSERTAR_UNO;

| ACTUALIZAR_UNO;

| ELIMINAR_UNO;

| BUSCAR_TODO;

| BUSCAR_UNO;

c. ***CREAR_BASE → CrearBD ID = nueva CrearBD()***

d. ***ELIMINAR_BASE → EliminarBD ID = nueva EliminarBD()***

e. ***CREAR_COLL → CrearColeccion ID = nueva CrearColeccion()***

f. ***ELIMINAR_COLL → EliminarColeccion ID = nueva EliminarColeccion(STRING)***

g. ***INSERTAR_UNO → InsertarUnico ID = nueva InsertarUnico(STRING, STRING)***

h. ***ACTUALIZAR_UNO → ActualizarUnico ID = nueva ActualizarUnico(STRING, STRING)***

i. ***ELIMINAR_UNO → EliminarUnico ID = nueva EliminarUnico(STRING)***

j. ***BUSCAR_TODO → BuscarTodo ID = nueva BuscarTodo(STRING)***

k. ***BUSCAR_UNO → BuscarUnico ID = nueva BuscarUnico(STRING)***