Universidad De San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas

Lenguajes formales y de programación
Sección “A-”

“MANUAL TÉCNICO”

Samuel Alejandro Pajoc Raymundo.

Carné: 201800665

Objetivos

General:

Brindar al lector una guía que contenga la información del manejo de clases, atributos, métodos y del desarrollo de los menús, y así facilitar futuras actualizaciones o futuras modificaciones realizadas por terceros.

Específicos:

- Mostrar al lector una descripción lo más completa y detallada posible del SO, IDE entre otros utilizados para el desarrollo de la aplicación.

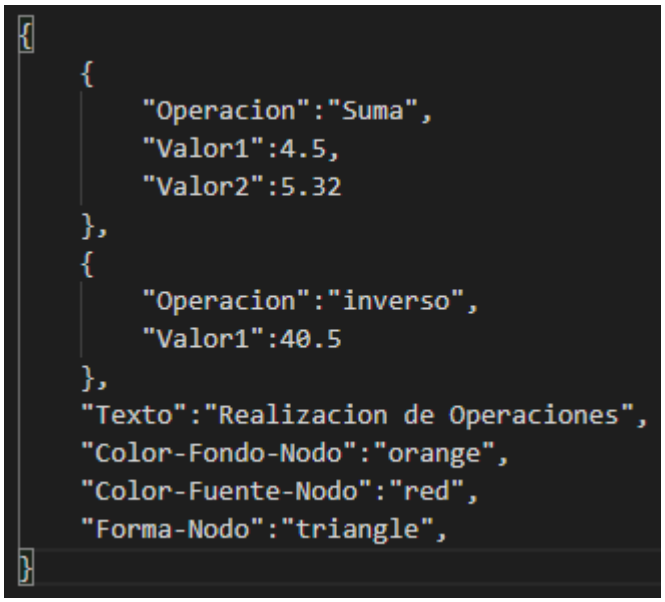
- Proporcionar al lector una explicación técnica/formal de los procesos y relaciones entre métodos y atributos que conforman la parte operativa de la aplicación.

Introducción

Este manual técnico tiene como finalidad dar a conocer al lector, una idea base para que pueda realizar futuras actualizaciones al software, indicando el IDE utilizado para su creación, su versión, requerimientos del sistema, etc…

La aplicación tiene como objetivo analizar el contenido de un archivo de texto plano, inicialmente cargado por el usuario en su correspondiente menú, con formato (flp), además, proporcionar de una manera más amigable al usuario, la detección de errores que contenga dicho archivo en su escritura; la aplicación fue creada con el lenguaje de python Version 3.11.1 (64-bits).

Ejemplo del archivo de entrada:



El formato del archivo es:

[a-z][A-Z] : letras de la A - Z, mayúsculas y minúsculas. Permitiendo formar palabras.

[0-9] : Números del 0 al 9. Permitiendo formar números enteros o decimales.

‘ . ‘ : Punto, indica el inicio de los decimales.

{ , } : Llaves, indican el inicio o cierre de una operación.

“ “ : comillas dobles, indican el inicio de una instrucción, valor o propiedad.

‘:’ : Dos puntos, indican el inicio del valor que acompaña a un valor o propiedad previamente establecida.

[,] : Corchetes, indica el inicio de una lista, la cual puede contener una operación y N valores.

‘ – ‘ : Guion, indica el espacio de separación en una característica específica.

- Si este formato no se respeta, el programa dará un mensaje de error, dado que el compila miento no estaría completo.

- Además, si el contenido del archivo posee caracteres no permitidos, el programa indicara su ubicación (columna, fila), de tal forma que este sea removido o sustituido por uno valido.

Descripción de la Solución

Esta aplicación fue desarrollada aplicando diversos paradigmas de programación como programación orientada a objetos (POO) y programación estructural.

- Se desarrolló distintos tipos de clases para poder tener un mejor control del análisis del programa o de los mismos datos obtenidos en el archivo cargado. Las clases que se crearon fueron:
    - Clase analizarT: Esta clase contiene el autómata finito determinista, el cual permite analizar carácter por carácter, y así obtener los datos más esenciales del archivo previamente cargado.
    - se hace un llamado a travez de un objeto creado en el menú principal llamado “analizar archivo”. Un ejemplo de esta clase es “analizarArchivo.AnalizarEntrada(texto)”.
    - Clase operaciones: esta clase permite realizar todas las operaciones matemáticas, y al finalizar, retorna una lista con cada valor leído, junto a su operación y su correspondiente resultado.
    - Clase mostrarError: esta clase permite la elaboración de un archivo de texto plano con formato (flp) y por medio de python, la escritura del archivo de salida en formato JSON, los datos utilizados para esta operación, se reciben después de haber analizado el archivo de entrada e infortunadamente no compilados ya que posee errores en su escritura.
    - Clase main: esta clase contiene todos los elementos que permiten crear el entorno grafico del programa, además contiene sub-métodos los cuales le dan un funcionamiento correcto a todos los botones y entradas de texto ubicado en cada ventana desplegable.

- Clase grafo: Esta clase permite la elaboración del grafo (desarrollado en graphviz, pero a su vez implementando un ambiente lógico por medio del lenguaje de python). Esta clase también cuenta con una función definida, la cual escribe un archivo en formato dot. Y luego solicita abrirlo, para que de este modo, el usuario pueda visualizar su resultado.

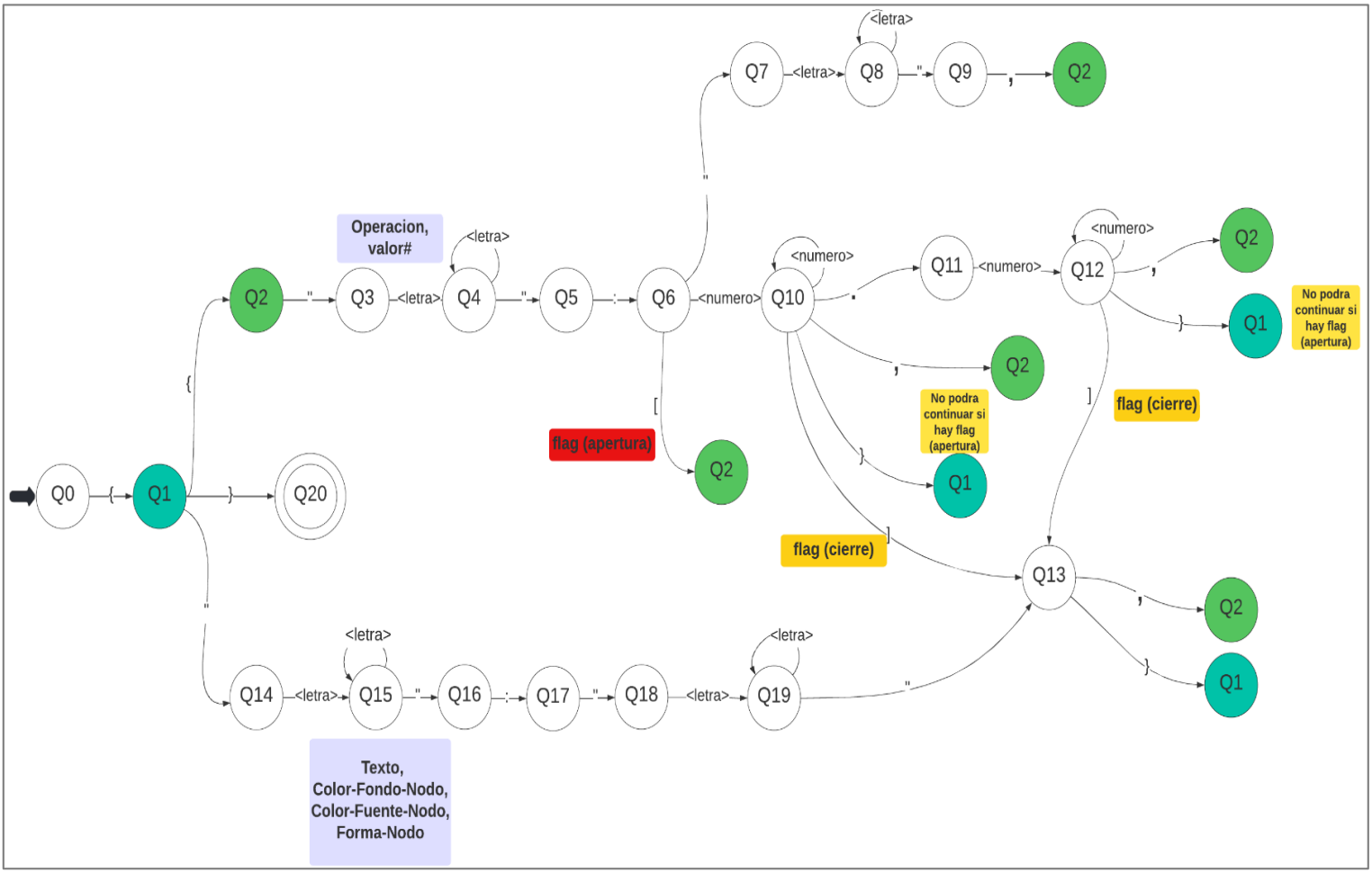
    - grafo(self, datos, instrucciones)
    - esta función recibe como paramentos los valores de los números junto a sus resultados y también recibe las especificaciones para elaborar el grafo, este grafo se finaliza en formato dot.png.

  Para poder almacenar el archivo en formato dot por medio de python, se utilizó la librería render, la cual permite crear la imagen png a travez del archivo escrito dot.

  Para poder abrir el archivo (imagen en formato png), se utilizó la función "webbrowser" y se colocó un alias de la misma como "wb", y por medio de su función "wb.open_new(>>nombre_archivo.png<<)", es que se logra desplegar la imagen generada-

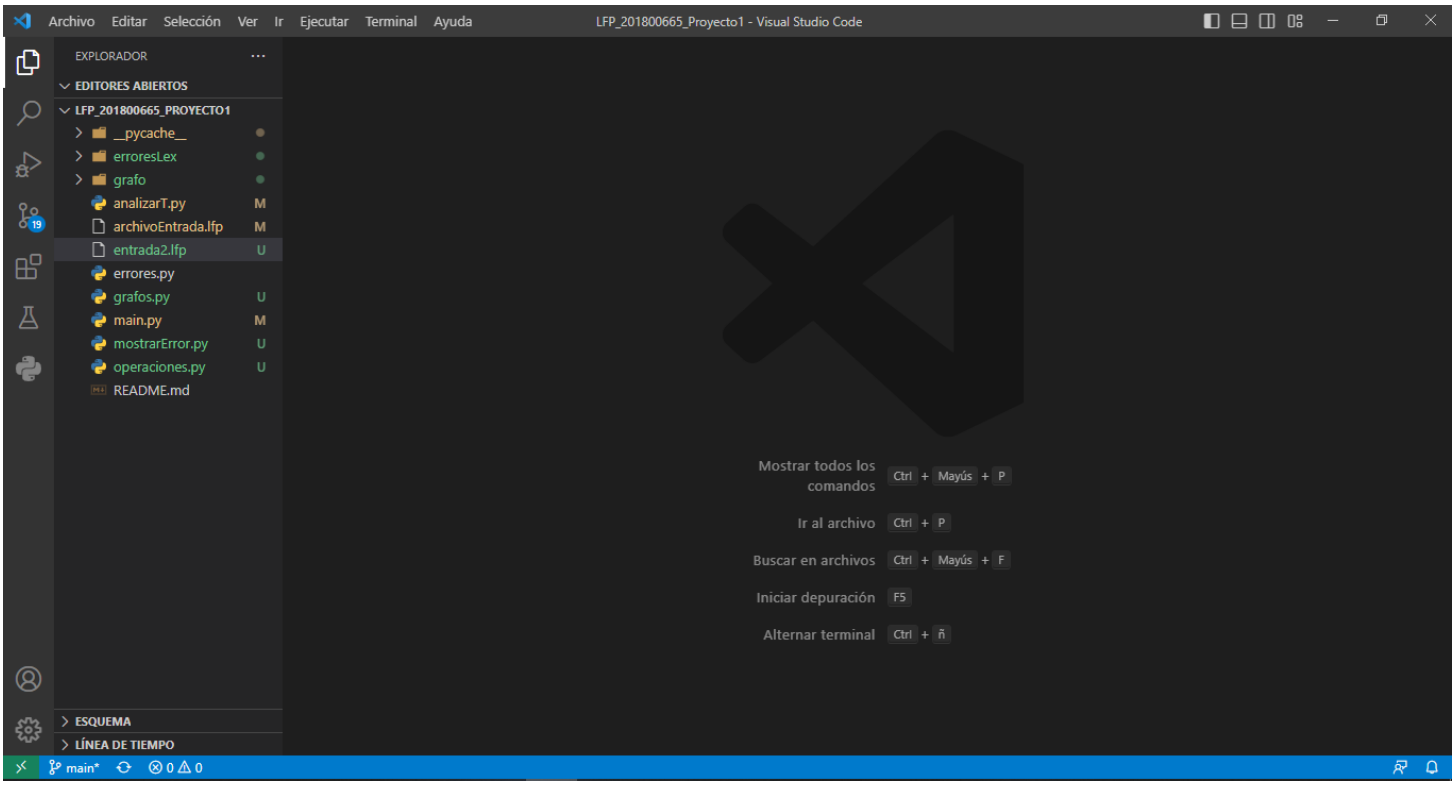
- Cada vez que se realice una acción importante o ocurra un problema, el programa le indicara el estado del programa, por medio de pequeños mensajes desplegados en ventanillas llamadas "mesagebox", se implementó dos tipos de advertencia para mostrar los estados, los cuales son :
    - .showerror
    - .showinfo

AFD (Autómata Finito Determinista)



IDE

El IDE con el que se desarrolló el proyecto fue Visual Studio Code.



Librerías Utilizadas

Las librerías utilizadas para el desarrollo de este proyecto fueron:

```
import tkinter
import tkinter.font as tkFont
import easygui
from tkinter import DISABLED, END, NORMAL, messagebox
from analizarT import analizar
from operaciones import Operacion
from grafos import CrearGrafo
from mostrarError import mostrarErroresLex

import webbrowser as wb
```