

# Manual Técnico

## Proyecto 1



Lenguajes Formales y de Programación  
Francisco Daniel Peruch de León  
Carné: 202100639

En este documento se presenta los detalles técnicos de la aplicación, el cual fue desarrollado en el lenguaje Python.

Se crearon diferentes ficheros de Python para poder crear y manejar las diferentes funcionalidades de la aplicación, los cuales se detallan a continuación:

### **Ventana:**

En este fichero se desarrollaron las diferentes funciones para poder crear la interfaz gráfica de la aplicación, la cual fue desarrollada utilizando la librería Tkinter.

Usando las distintas funciones que provee la librería Tkinter se crearon distintos botones y área de texto que fueron de utilidad para indicar al usuario de una manera agradable las funciones de la aplicación. Las funciones creadas en este fichero son las siguientes:

- *analizar()*: Esta función fue utilizada para poder correr el fichero llamado “analizador” y fue utilizado en el botón llamado “botón\_analizar” como el parámetro command.
- *generarArchivoJson()*: Esta función fue utilizada para poder correr el fichero llamado “generarJson” y fue utilizado en el botón llamado “botón\_errores” como el parámetro command.
- *ventana\_guardar()*: Esta función fue utilizada para generar una ventana secundaria llamada “ventana2” y fue utilizado en el botón llamado “botón\_errores” como el parámetro command.
- *guardar\_archivo()*: Esta función fue utilizada para poder recibir el texto a analizar de un área de texto, generando un archivo .txt el cual contenía el texto introducido en el área de texto y fue utilizado en el botón llamado “botón\_guardar1” como el parámetro command.

### **Token:**

En este fichero se creó una clase llamada “Token”, la cual contiene los siguientes atributos:

- *fila*: Guarda la fila en donde se encontró el carácter.
- *columna*: Guarda la columna en donde se encontró el carácter.
- *Lexema*: Guarda el carácter encontrado.

## **Analizar:**

Este fichero tiene la funcionalidad de iniciar el análisis léxico del texto contenido en el archivo .txt que se esta analizando, en este archivo se importó todo lo que contenía el fichero llamado “analizadorPrueba” y el fichero llamado “operaciones”. Las funciones que este fichero contiene son las siguientes:

- Comenzar(): Esta función permite crear un objeto de la clase “Automata”, dicha clase es importada del fichero “analizarPrueba”, por medio de este objeto se llama a la función analizar(). Para los parámetros requeridos para la función analiza() se envió una variable llamada “cadena”, la cual contiene el texto del archivo .txt que se esta analizando y un objeto de la clase “Operacion” que se genera por defecto.

La lista que la función analizar() regresa es guardada en la variable resultado la cual es iterada en un ciclo for par poder realizar las operaciones contenidas dentro de esta.

## **operaciones:**

Este fichero contiene una clase llamada “Operacion” la cual tiene una función y dos atributos que se detallan a continuación:

- tipo: En este atributo se guarda el tipo de operación que se desea realizar.
- operandos: Este atributo es una lista que comienza vacía, pero en el trascurso de la ejecución del programa se van guardando diferentes valores u objetos de la clase “Operacion” que contendrán a su vez los valores necesarios para la realización de cada operación.
- operar(): En esta función se realizar el calculo de las distintas operaciones indicadas en el archivo .txt que se está analizando.

Dichas operaciones son las siguientes:

- ✓ Suma
- ✓ Resta
- ✓ Multiplicación
- ✓ División
- ✓ Potencia
- ✓ Inverso
- ✓ Seno
- ✓ Coseno
- ✓ Tangente

- ✓ Mod
- ✓ Raíz

### **generarJson:**

En este se importa la lista llamada “tabla\_errores\_total” del fichero “archivoErrores” el cual contine objetos de la clase “Errores” del fichero “Errores”. Esta lista es utilizada para crear un archivo .json el cual contendrá la información de un error encontrado durante el análisis de el texto contenido en el archivo analizado.

### **Errores:**

En este fichero se encuentra una clase llamada “Errores”, la cual tiene los siguientes atributos que serán útiles para crear objetos que guarden la información de un error encontrado durante el análisis correspondiente:

- fila: En este atributo se guarda la fila donde ocurrió el error.
- columna: En este atributo se guarda la columna donde ocurrió el error:
- lexema: En este atributo se guarda el carácter que fue un error en el análisis.
- id: En este se guarda el número del identificador correspondiente al error encontrado.

### **archivoErrores:**

En este archivo se obtiene la variable tabla\_errores, la cual es proveniente desde el fichero “analizadorPrueba”, dicho valor es una lista la cual se guarda en otra lista llamada “tabla\_errores\_total”.


### **analizadorPrueba:**


En este fichero se realiza el análisis léxico al texto que contiene el archivo .txt que se esta analizando. Para realizar el análisis se necesito la implementación de una autónoma (Figura 1), la cual fue utilizada para poder analizar cada carácter dentro del texto y revisar si correspondía a un patrón en cierto estado. De esta manera se verificaba si se encontraba todo en orden con esa parte del texto analizado para poder realizar la operación que indicaba el texto del archivo analizado.

En este fichero se encuentran la siguiente clase junto con sus funciones y atributos:

- letras: En este atributo se guardan los caracteres aceptados para un token del tipo letras.
- numeros: En este atributo se guardan los caracteres aceptados para un token de tipo numérico.
- cadena: En este atributo se guarda la parte actual del texto del archivo .txt que está siendo analizado.
- fila: En este atributo se guarda la fila donde se encuentra el carácter analizado.
- columna: En este atributo se guarda la columna donde se encuentra el carácter analizado.
- estado actual: En este atributo se guarda el estado donde se encuentra la autónoma en el transcurso de la ejecución del programa.
- estado anterior: En este atributo se encuentra el estado que es anterior al estado donde se encuentra actualmente la autónoma.
- estado aceptacion: En este atributo se guarda el estado que se considera de aceptación.
- tabla tokens: Este atributo es una lista que guarda los objetos de la clase "Token" que guarda la información del lexema encontrado.
- tabla errores: Este atributo es una lista que guarda los objetos de la clase "Errores" que guarda la información de los errores encontrados en la ejecución.
- analizar(): Esta función realiza la implementación de la autónoma utilizada para el análisis del texto contenido el archivo.txt.
- guardar tokens(): En esta función se crean los diferentes objetos de la clase "Token" y se guardan en la lista "tabla\_tokens".
- guardar errores(): En esta función se crean los diferentes objetos de la clase "Errores" y se guardan en la lista "tabla\_errores".
- obtener tabla errores(): Esta función es de utilidad para retornar la tabla\_errores que será utilizada en otros ficheros para determinadas funcionalidades.

## Gramática Regular Utilizada:

 **Alfabeto**=Terminales{  
letras={a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z},numeros={1,2,3,4,5,6,  
7,8,9,0}}

 **NoTeminales**={0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,  
23,24,25,26,27}

 **Estado Inicial**:0

 **Estado de Aceptación**:9

 **Transiciones**:

- 0->{1
- 1->"2
- 1->{10
- 2->letras3
- 3->"4
- 3->letras3
- 4->:5
- 5->"6
- 6->letras7
- 7->letras7
- 7>"8
- 8->}9
- 8->,1
- 10->"11
- 11->letras12
- 12->"13
- 12->letras12
- 13->:14
- 14->"15
- 15->letras16
- 16->letras16
- 16->"17
- 17->,18
- 18->"19
- 19->letras20
- 20->letras20
- 20->"21
- 21->:22
- 22->[25
- 22->numeros23

- 23->.24
- 23->numeros23
- 23->]27
- 23->}26
- 23->,18
- 24->numeros24
- 24->]27
- 24->,18
- 24->}26
- 25->"11
- 27->]27
- 27->,18
- 27->}26
- 26->}9
- 26->,1

AFD

