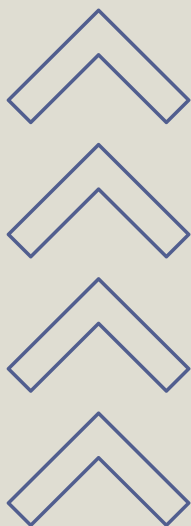


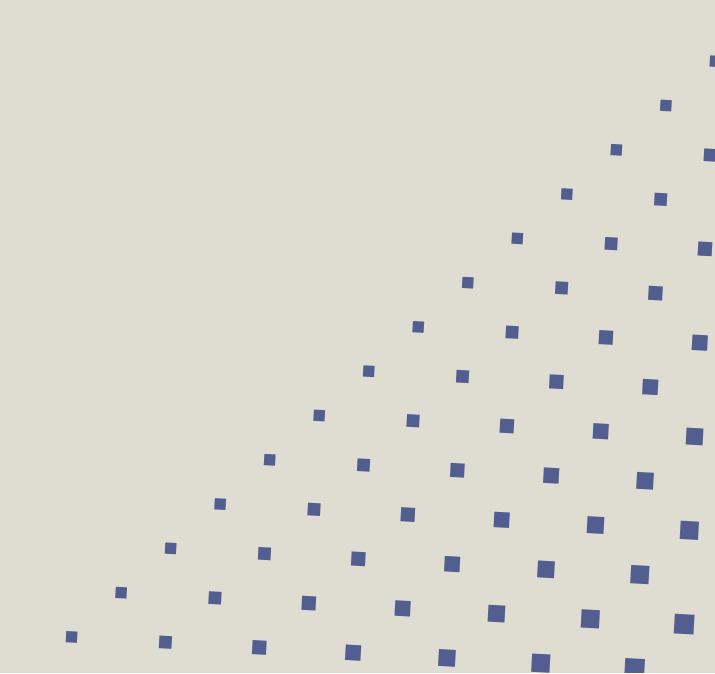


**LENGUAJES FORMALES DE  
PROGRAMACIÓN**

# **MANUAL TÉCNICO**




Informe redactado por:  
Jose Fuentes  
202001228





# ÍNDICE

• Introducción	1
• Información sobre el programa	1
• Importación del programa	2
• Tokens	3
• Gramática	3
• Operaciones	4
• Errores	5
• Generar Reportes	5
• AFD	6





# INTRODUCCIÓN

Este es un programa realizado a fin de ayudar en el análisis léxico de documentos con formato xml. Tratando de identificar los errores y el mismo tiempo realizando las operaciones que se le indiquen en el mismo.



## IMFORME SOBRE EL PROGRAMA

Este programa fue realizado en el lenguaje de Python versión 3.10 con la cual se desarrollo un software que ayudara al estudiante a saber como va en la carrera mediante los cursos aprobados, cursando y pendientes que tiene el estudiante. Por medio de agregación de cursos y con una carga masiva para facilitar el ingreso de cada uno de los cursos.

Este es un sistema que 64 bits, se puede utilizar en los siguientes sistemas operativos:

- Windows 7
- Windows 8
- Windows 8.1
- Windows 10
- Windows 11

Requisitos para el uso del programa:

- 1 GB de ram
- Espacio en disco duro de 256 MB
- Procesador: Minimo Pentium hasta gama alta de procesadores



# IMPORTACIONES DEL PROGRAMA

Al iniciar la programación de este programa, se realizando varias importaciones para el mejor manejo del código e interacción con el usuario.

1. Las primeras 7 importaciones tiene que ver con el analizador léxico y las operaciones, errores que se manejaran a la hora de la lectura del archivo ingresado por el usuario.
2. La importación de tkinter es para la parte grafica, la cual ayudara para la interacción con el usuario y mejorar su experiencia.
3. La importación de "webbrowser" es para abrir un documento con una ruta ya predefinida.

```
from texto import Texto
from numero import Numero
from aritmeticas import Aritmeticas
from operador import Operador
from errores import Errores
from estilo import Estilo
from funcion import Funcion
from tkinter import *
from tkinter import filedialog
from tkinter.filedialog import askopenfilename
from tkinter import messagebox
import webbrowser as wb
```

## »»»» TOKENS

En esta parte del código se declaran los tokens que se utilizarán para la lectura del archivo de entrada, el cual contendrá diferentes signos y palabras claves las cuales servirán para desfragmentar el archivo, analizarlo y realizar las operaciones que se le indicaran.

Se mostrara una pequeña parte del código por cuestiones de estática.

```
tokens = (  
    'RESTILO',  
    'ROPERACIONES',  
    'RTIPO',  
    'RTEXT0',  
    'RTIPO2',  
    'RTEXT02',  
    'RFUNCION',
```

## »»»» GRAMÁTICA

Acá se examinara la parte gramática de cada uno de las números, cadenas de texto que se ingresaran para saber de que tipo es: entero, decimal, cadenas de texto, saltos de línea, espacios, etc.

Se dejara el código de un numero decimal.

```
def t_DECIMAL(t):  
    r'\d+\.\d+'  
    try:  
        t.value = float(t.value)  
    except ValueError:  
        print("Float value too large %d", t.value)  
        t.value = 0  
    return t
```



# OPERACIONES

En el archivo .py llamado aritmetica estan todas las operaciones ha realizar en el archivo que se le ingresara; en el archivo operador.py contiene la lista de operaciones que se pueden realizar tomando un valor especifico cada una de ellas.

Por motivo de estética no se mostrara todo el código para realizar la suma de cada numero que se ingresara, solamente se estará mostrando el operador suma y resta. Y también el código de operador con el valor asignado a cada una de las operaciones a realizar.

```
if self.tipo == Operador.SUMA:
    return generador.addExpresion(izq, der, '+') if getER else izq+der
elif self.tipo == Operador.RESTA:
    return generador.addExpresion(izq, der, '-') if getER else izq-der
```

```
class Operador(Enum):
    OPERACION = 0
    SUMA = 1
    RESTA = 2
    MULTIPLICACION = 3
    DIVISION = 4
    POTENCIA = 5
    MODULO = 6
    INVERSO = 7
    TANGENTE = 8
    COSENO = 9
    SENO = 10
    RAIZ = 11
```

## ERRORES

Al ingresar el archivo de entrada se podrá ir almacenando todos los errores que contendrá, tomando en cuenta que se tomaran de que tipo es, en que fila y columna se encuentran y cual es el error que identifico el programa.

```
def __init__(self, lexema, tipo, columna, fila):  
    self.lexema = lexema  
    self.tipo = tipo  
    self.columna = columna  
    self.fila = fila
```

## GENERAR REPORTES

Por medio de variables globales se podrá ir almacenando tanto los errores como las operaciones y el resultado de las mismas. Por medio de la función .write que se le hace a un archivo podemos ir agregando el código html que se necesitara para realizar el archivo html. Por motivos de estética solo se mostrará una imagen donde se puede agregar las operaciones y el resultado de la misma en una tabla.

```
for operacion in operaciones:  
    archivo.write('<tr>')  
    archivo.write(' <td>' + str(operacion[0]) + '</td>')  
    archivo.write(' <td>' + str(operacion[2]) + '</td>')  
    archivo.write('<tr>\n')
```

5

