202111718

Manual Técnico

Walter Javier Santizo Mazariegos

walterjav19@gmail.com

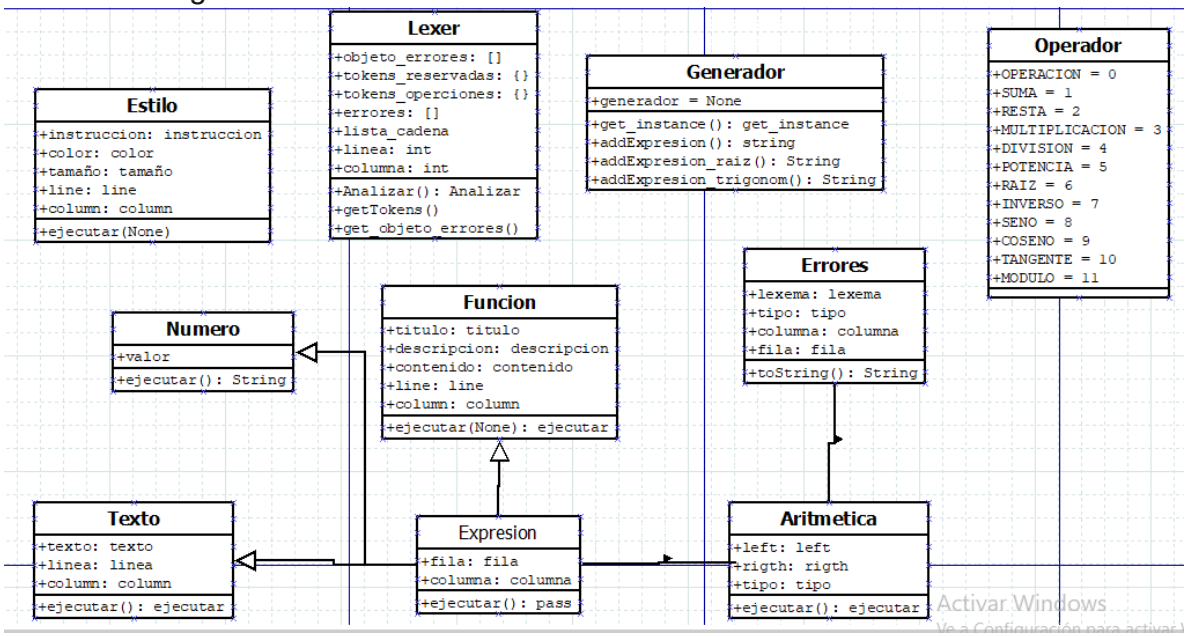
programación Orientada a objetos

La programación orientada a objetos se basa en el concepto de crear un modelo del problema de destino en sus programas. La programación orientada a objetos disminuye los errores y promociona la reutilización del código. Python es un lenguaje orientado a objetos. Los objetos definidos en Python tienen las características siguientes

Clases utilizadas

- Análisis Léxico
- Aritméticas
- Errores
- Estilo
- Expresión
- Función
- Generador
- Numero
- Operador
- Texto

Se utilizaron estas clases con sus atributos valiéndose de las propiedades para crear un lexer con este paradigma de programación y se modelo u diagrama de clase de la siguiente forma



```
class Lexico:
    angular_abierta = 0
    igual = False

    def __init__(self,lista_cadena):
        self.objeto_errores=[]
        self.tokens_reservadas = {
            'Tipo'      : 't_TIPO',
            'Operacion' : 't_OPERACION',
            'Numero'    : 't_NUMERO',
            '<'         : 't_AA',
            '>'         : 't_AC',
            '='         : 't_IGUAL',
            '/'         : 't_DIV',
        }

        self.tokens_operaciones = {
            'SUMA'    : 't_SUMA',
            'RESTA'   : 't_RESTA',
            'MULTITPLICACION'    : 't_MULT',
            'DIVISION'    : 't_DIV',
        }
        self.errores = ['?',"$","~","!","¡","%","#","(",")",":",",","-","|","{","}","[","]","_","^","¨"]
        self.lista_cadena=lista_cadena
        self.linea = 1
        self.columna = 0
```


Inicializamos todos los tokens que vamos a usar y palabras que ya están reservadas en nuestro lenguaje y los elementos como los apuntadores de fila y columna que se usaran

```
class Aritmeticas(Expression):

    def __init__(self, left, right, tipo, fila, column):
        self.left = left
        self.right = right
        self.tipo = tipo
        super().__init__(fila, column)
```

En aritméticas se procesara los tokens que lee el analizador con el fin de ejecutarlos de forma recursiva y heredando de la clase expresión que obtiene las cadenas de texto y de todas las regex

```
class Errores():

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

Almacenara un objeto del tipo error para posteriormente generar un reporte

```python
from abc import ABC, abstractmethod

class Expression(ABC):

    def __init__(self, fila, column):
        self.fila = fila
        self.column = column

    @abstractmethod
    def ejecutar(self, getER):
        pass
```

Obtenemos el char con el cual vamos a generar las expresiones

```python
from expression import *
class Numero(Expression):

    def __init__(self, valor, fila, column):
        self.valor = valor
        super().__init__(fila, column)

    def ejecutar(self, getER):
        return self.valor
```

Genera los números que posteriormente se van operar mediante el operador

```python
from enum import Enum

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

Enumeramos las opciones que podemos tener en nuestro lenguaje

programación Estructurada

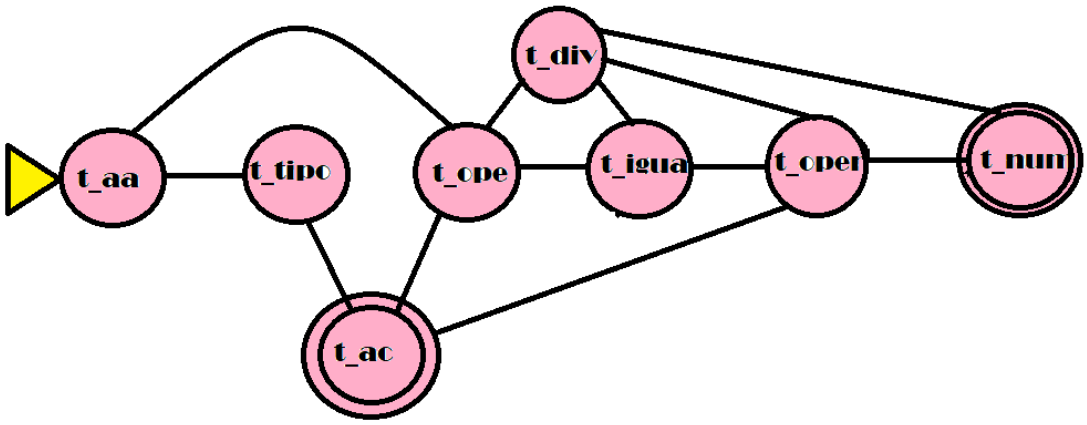
De esta forma se modelo la parte grafica o frontend de la aplicación por medio de métodos, ciclos y bucles que permitieran manipular el usuario de una manera sencilla

autómata finito deterministas

primero determinamos el alfabeto de nuestro alfabeto y los tokens que serán palabras reservadas

$$\sum = \{t_{suma}, t_{resta}, t_{multiplicacion}, t_{division}, t_{modulo} \; t_{potencia}, t_{seno}, t_{coseno}, t_{tangente}, t_{raiz}\}$$

$$\sum = \{t_{tipo}, t_{operacion}, t_{numero}, t_{aa}, t_{ac}, t_{igual}, t_{div}\}$$



Tokens en Python

```
tokens = (
    'RESTILO',
    'ROPERACIONES',
    'RTIPO',
    'RTEXTO',
    'RTIPO2',
    'RTEXTO2',
    'RFUNCION',
    'RTITULO',
    'RDESCRIPCION',
    'RCONTENIDO',
    'ROPERACION',
    'RCOLOR',
    'RTAMANIO',
    'RNUMERO',
    'RSUMA',
    'RRESTA',
    'RMULTIPLICACION',
    'RDIVISION',
    'RRAIZ',
    'RPOTENCIA',
    'RSENO',
    'RCOSENO',
    'RTANGENTE',
    'RMOD',
    'RINVERSO',
    'RESCRIBIR',
    'LLAA',
    'LLAC',
    'IGUAL',
    'DIV',
    'ENTERO',
    'DECIMAL'
```

```
t_RESTILO = r'Estilo'
t_RTIPO2  = r'TIPO'
t_RTEXTO2  = r'TEXTO'
t_RTIPO       = r'Tipo'
t_RTEXTO        = r'Texto'
t_RFUNCION      = r'Funcion'
t_RTITULO       = r'Titulo'
t_RDESCRIPCION  = r'Descripcion'
t_RCONTENIDO    = r'Contenido'
t_ROPERACION     = r'Operacion'
t_ROPERACIONES = r'Operaciones'
t_RCOLOR        = r'Color'
t_RTAMANIO      = r'Tamanio'
t_RSUMA       = r'SUMA'
t_RRESTA       = r'RESTA'
t_RMULTIPLICACION = r'MULTIPLICACION'
t_RDIVISION = r'DIVISION'
t_RRAIZ=r'RAIZ'
t_RMOD=r'MOD'
t_RSENO=r'SENO'
t_RPOTENCIA=r'POTENCIA'
t_RCOSENO=r'COSENO'
t_RTANGENTE=r'TANGENTE'
t_RINVERSO = r'INVERSO'
t_RESCRIBIR = r'ESCRIBIR'
t_RNUMERO     = r'Numero'
t_LLAA        = r'<'
t_LLAC        = r'>'
t_IGUAL       = r'='
t_DIV         = r'/'
t_CORA        = r'\['
t_CORC        = r'\]'
t_AZUL        = r'AZUL'
```