Manual Tecnico

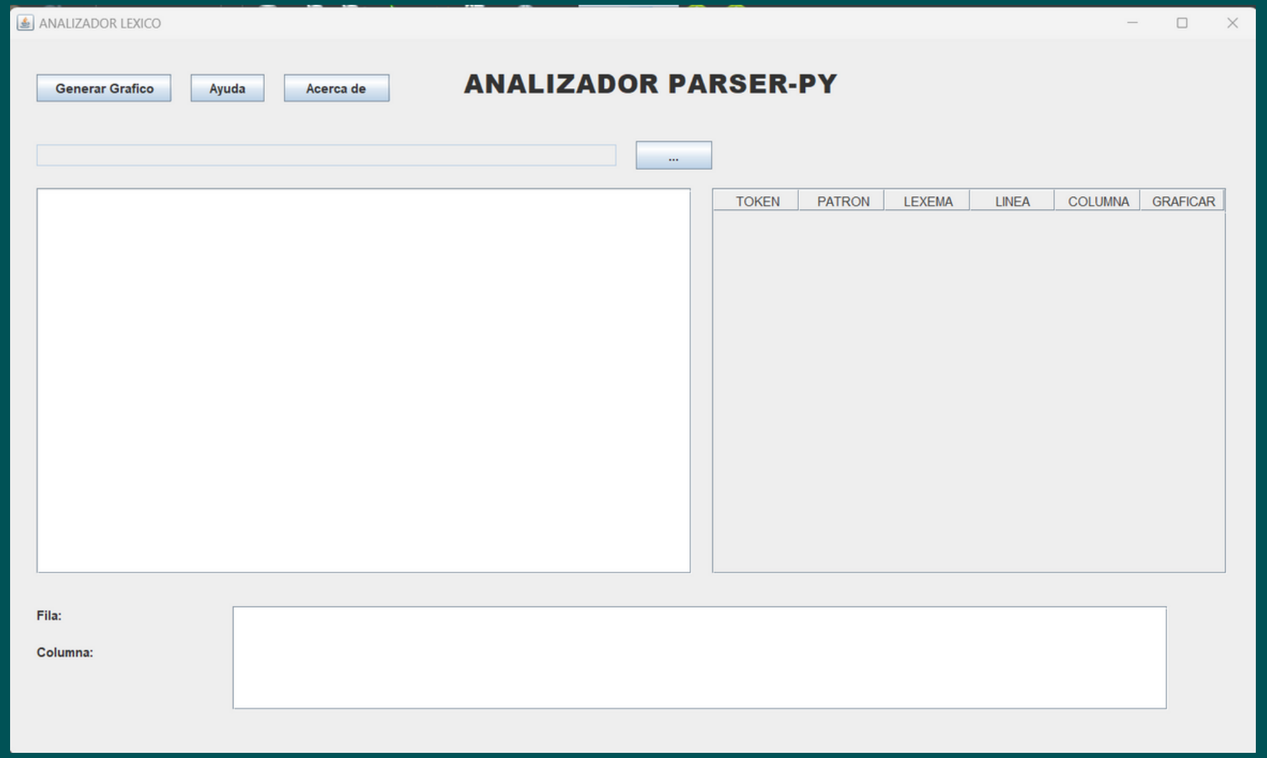
Analizador Lexico - Lenguajes Formales y de Programacion

Enmer Oswaldo Sandoval Mazariegos 202031773

REQUISITOS:

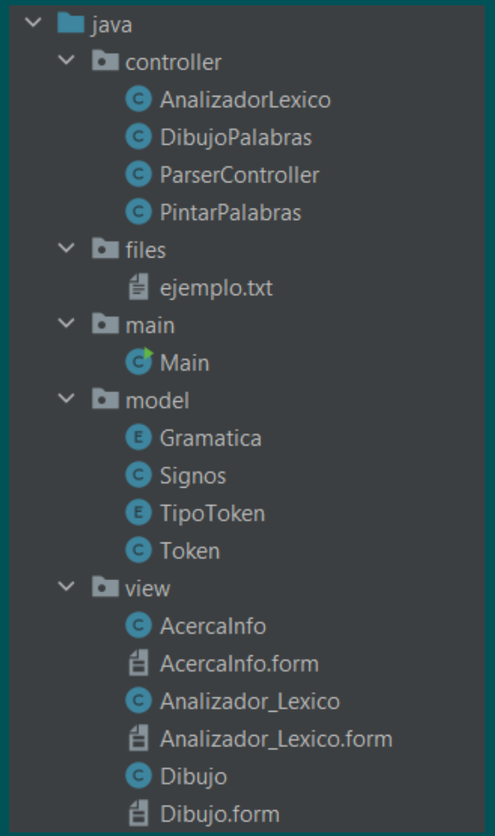
- JDK 17 instalado
- Intel celeron o superior
- 2gb de ram
- 50MB de espacio

Interfaz Grafica



Creada desde un JFrame usamos un JTextPane para ingresar el texto, usamos un JFileChooser y tambien una JTable para poder ver los lexemas generados, tenemos un JTextArea para ver la informacion de lexemas

JFileChooser



Clases usadas

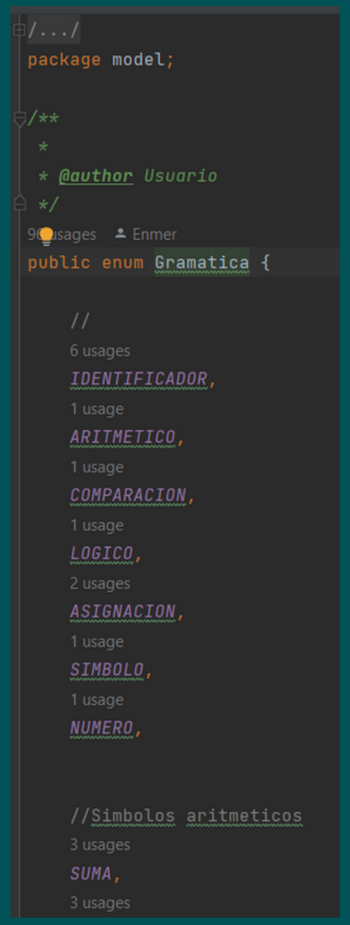
Analizador

```
1 usage  Enmer
public AnalizadorLexico(List<Token> tokens) { this.tokens = tokens; }

1 usage  Enmer
public void analizandoTextArea(JTextPane textPane, JTextArea areaDeTexto, String resultados, JTable table) {
    tokens = analizar(textPane, resultados);
    for (int i = 0; i < tokens.size(); i++) {
        String token = tokens.get(i).toString();

        resultados += token + "\n";
        areaDeTexto.setText(resultados);
    }
    llenadoTable(table, tokens);
}

1 usage  Enmer
public List<Token> analizar(JTextPane textPane, String resultados) {
    String textAreaString = textPane.getText();
    List<Token> tokens = new ArrayList<>();
    StringBuilder palabra = new StringBuilder();
    Gramatica tipoActual = null;
    // Agregamos una variable para controlar el comportamiento del '#'

    for (int i = 0; i < textAreaString.length(); i++) {
        char c = textAreaString.charAt(i);
        String palabraString = palabra.toString();

        if (inicioPalabra(c)) {
            if (palabraString.startsWith("\"") && !palabraString.endsWith("\"")) {
                c = textAreaString.charAt(i + 1);
```

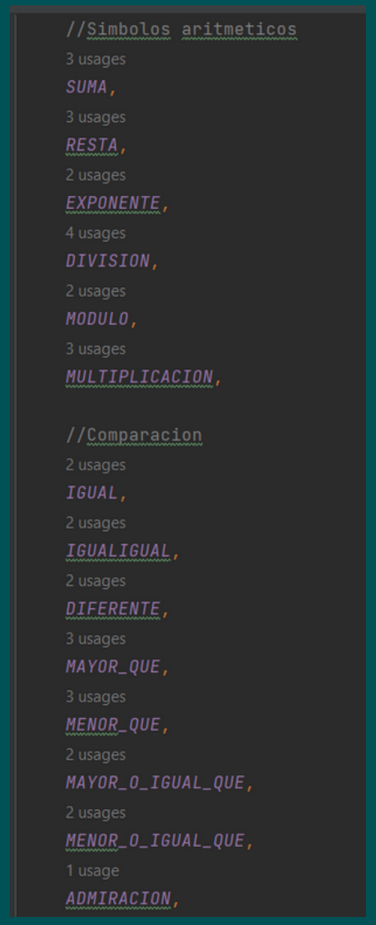
Analizador

```
        if (inicioPalabra(c)) {
            if (palabraString.startsWith("\"") && !palabraString.endsWith("\"")) {
                c = textAreaString.charAt(i + 1);
                palabra.append(c);
                procesarPalabra(tokens, palabra.toString(), tipoActual);
                palabra.setLength(0);
                tipoActual = null;
            } else {
                if (tipoActual == Gramatica.COMENTARIO) {
                    if (c == '\n' || c == '\r' || c == '\t') {
                        procesarPalabra(tokens, palabra.toString(), tipoActual);
                        palabra.setLength(0);
                        tipoActual = null;
                    }
                } else if (tipoActual != Gramatica.COMENTARIO) {
                    procesarPalabra(tokens, palabra.toString(), tipoActual);
                    palabra.setLength(0);
                    tipoActual = null;
                }
            }
        } else {
            palabra.append(c);
            tipoActual = determinarTipo(palabra.toString(), tipoActual);
        }
    }

    return tokens;
}

1 usage  Enmer
```

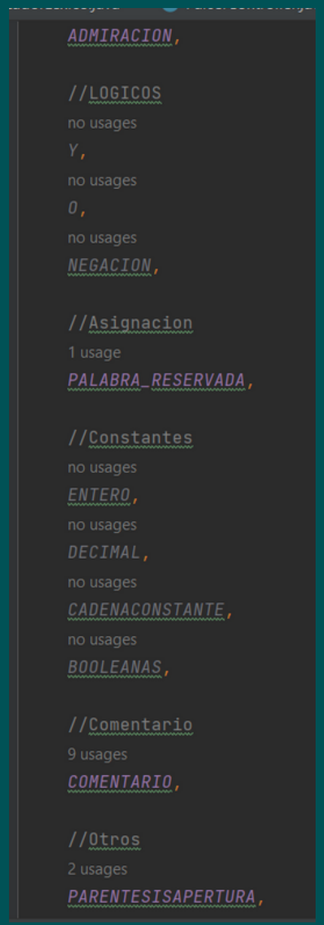
ENUM



```
/.../
package model;

/**
 *
 * @author Usuario
 */
9 usages  Enmer
public enum Gramatica {

    //
    6 usages
    IDENTIFICADOR,
    1 usage
    ARITMETICO,
    1 usage
    COMPARACION,
    1 usage
    LOGICO,
    2 usages
    ASIGNACION,
    1 usage
    SIMBOLO,
    1 usage
    NUMERO,


    //Simbolos aritmeticos
    3 usages
    SUMA,
    3 usages
```



```
    //Simbolos aritmeticos
    3 usages
    SUMA,
    3 usages
    RESTA,
    2 usages
    EXPONENTE,
    4 usages
    DIVISION,
    2 usages
    MODULO,
    3 usages
    MULTIPLICACION,

    //Comparacion
    2 usages
    IGUAL,
    2 usages
    IGUALIGUAL,
    2 usages
    DIFERENTE,
    3 usages
    MAYOR_QUE,
    3 usages
    MENOR_QUE,
    2 usages
    MAYOR_O_IGUAL_QUE,
    2 usages
    MENOR_O_IGUAL_QUE,
    1 usage
    ADMIRACION,
```



```
    ADMIRACION,

    //LOGICOS
    no usages
    Y,
    no usages
    O,
    no usages
    NEGACION,

    //Asignacion
    1 usage
    PALABRA_RESERVADA,

    //Constantes
    no usages
    ENTERO,
    no usages
    DECIMAL,
    no usages
    CADENACONSTANTE,
    no usages
    BOOLEANAS,

    //Comentario
    9 usages
    COMENTARIO,

    //Otros
    2 usages
    PARENTESISAPERTURA,
```

```
    PARENTESISAPERTURA,
    2 usages
    PARENTESISCIERRE,
    2 usages
    LLAVEAPERTURA,
    2 usages
    LLAVECIERRE,
    2 usages
    CORCHETEAPERTURA,
    2 usages
    CORCHETECIERRE,
    2 usages
    COMA,
    2 usages
    PUNTO,
    2 usages
    PUNTO_Y_COMA,
    2 usages
    DOS_PUNTOS,
    1 usage
    COMILLAS,
    2 usages
    COMILLASDOBLES,
    2 usages
    BARRAINVERSA,

    //Numeros
    2 usages
    CERO,
    2 usages
    UNO,
```

CLASE TOKEN

```java
/.../
package model;

import model.Gramatica;

/**
 *
 * @author Usuario
 */
2 usages  Enmer
public class Token {
    3 usages
    private Gramatica token;
    4 usages
    private Gramatica gramatica;
    4 usages
    private String lexema;
    4 usages
    private int fila;
    4 usages
    private int columna;

    1 usage  Enmer
    public Token() {
    }

    5 usages  Enmer
    public Token(Gramatica token ,Gramatica gramatica, String lexema, int fila, int columna) {
        this.token = token;
        this.gramatica = gramatica;
        this.lexema = lexema;
```