



Alumno: Matuz Gómez Braulio Alanni

Materia: Seminario de Traductores de lenguajes 2

sección: D02 Código estudiante: 219293637

2024A

Título de la Tarea: Gramática del compilador

## Introducción

En esta fase se mostrará la extensión del léxico con el que trabajara el compilador que se ha creado como.

## Contenido

En cuanto a los avances se demuestra tener ya una interfaz gráfica para que se detecte el léxico permitido dentro del compilador creado para así dar cuentas de lo que se entiende y lo que no. en perspectiva se mostrará parte del código así como los resultados de la experimentacion.

```
class Token:
    def __init__(self, tipo, valor=None):
        self.tipo = tipo # Tipo del token (entero, real, operador, identificador, etc.)
        self.valor = valor # Valor del token

class AnalizadorLexico:
    def __init__(self, codigo_fuente):
        self.codigo_fuente = codigo_fuente # Código fuente a analizar
        self.patron_entero = r'\d+' # Patrón para reconocer enteros
        self.patron_real = r'\d+\.\d+' # Patrón para reconocer números reales
        self.patron_operadores = r'[+ \- * / =]' # Patrón para reconocer operadores
        self.patron_identificador = r'[a-zA-Z][a-zA-Z0-9_]*' # Patrón para reconocer identificadores
        self.caracteres_especiales = r'[\[\] \{ \} ! \\' # Caracteres especiales
        self.tokens = [] # Lista para almacenar los tokens encontrados
        self.errores = [] # Lista para almacenar los errores léxicos encontrados

    def analizar(self):
        pos = 0
        while pos < len(self.codigo_fuente):
            if self.codigo_fuente[pos].isspace():
                pos += 1
            elif self.codigo_fuente[pos].isdigit():
                match_entero = re.match(self.patron_entero, self.codigo_fuente[pos:])
                if match_entero:
                    valor = match_entero.group(0)
                    self.tokens.append(Token('entero', valor))
```

## los resultados

Analizador de Código

Introduce tu código:

```
main

int 8,9,7;
//
loop 7781

char A,B,C

float; sodj

+}´{-*]
```

Analizar código

Resultado del análisis:

Tokens encontrados:

Token inválido: main (Tipo: identificador)

Token inválido: int (Tipo: identificador)

Token inválido: 8 (Tipo: entero)

Token inválido: 9 (Tipo: entero)

Token inválido: 7 (Tipo: entero)

Token inválido: / (Tipo: operador)

Token inválido: / (Tipo: operador)

Token inválido: loop (Tipo: identificador)

Token inválido: 7781 (Tipo: entero)

Token inválido: char (Tipo: identificador)

Token inválido: A (Tipo: identificador)

Token inválido: B (Tipo: identificador)

Token inválido: C (Tipo: identificador)

Token inválido: float (Tipo: identificador)

Token inválido: sodj (Tipo: identificador)

Token inválido: + (Tipo: operador)

Token inválido: } (Tipo: caracter\_especial)

Token inválido: ´ (Tipo: caracter\_especial)

Token inválido: { (Tipo: caracter especial)

por el momento son pruebas mínimas pero factibles para que este mismo funcione

## Referencias

- <https://github.com/TraductoresLenguajes2/Traductores/tree/master/Modulo4>