ANALIZADOR SINTACTICO (OBJ)

OMAR ALEJANDRO QUIROZ TRUJILLO

- **DESARROLLO**

Un analizador sintáctico, también conocido como parser, es una parte crucial de los compiladores e intérpretes en la ciencia de la computación. Su función principal es analizar la estructura gramatical de un lenguaje de programación (o cualquier otro tipo de lenguaje formal) para determinar si cumple con las reglas sintácticas definidas por el lenguaje.

Aquí están algunas de las funciones principales de un analizador sintáctico:

Análisis de la estructura gramatical: El analizador sintáctico toma el código fuente como entrada y lo divide en una estructura jerárquica que refleje la gramática del lenguaje. Esta estructura puede ser representada mediante árboles sintácticos o mediante otras estructuras de datos.

Verificación de la sintaxis: El analizador sintáctico verifica si el código fuente se ajusta a las reglas sintácticas del lenguaje de programación en cuestión. Si encuentra un error sintáctico, generalmente produce un mensaje de error indicando dónde ocurrió el error y qué tipo de error es.

Producción de un árbol sintáctico o una representación interna: A partir del código fuente, el analizador sintáctico puede construir un árbol sintáctico que represente la estructura jerárquica del código. Este árbol es útil para las etapas posteriores del proceso de compilación o interpretación.

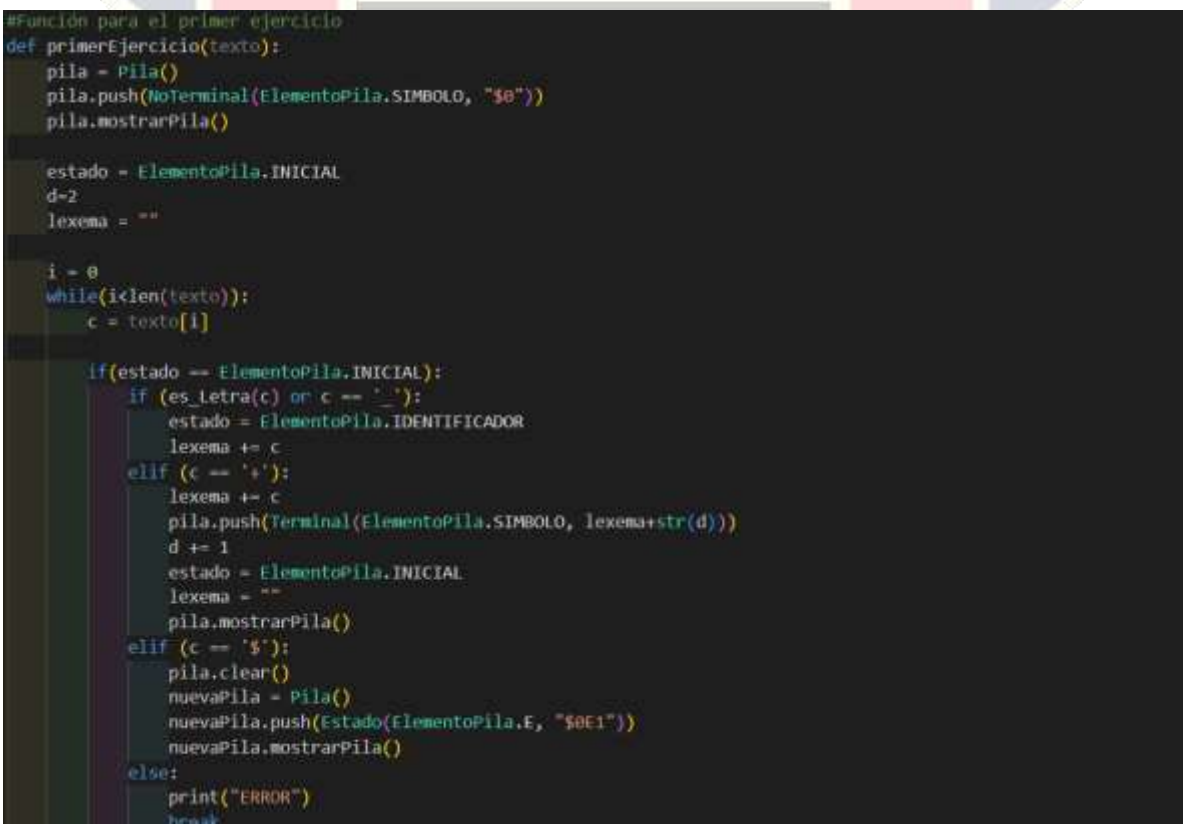
Preparación para la generación de código intermedio: En los compiladores, el analizador sintáctico es una etapa previa crucial antes de la generación de código intermedio. El árbol sintáctico o la representación interna generada por el analizador sintáctico sirve como entrada para la fase de generación de código.

Optimización sintáctica: En algunos casos, el analizador sintáctico puede aplicar optimizaciones a la estructura del código fuente para mejorar su rendimiento o su legibilidad, aunque este tipo de optimizaciones son menos comunes y suelen ser realizadas en etapas posteriores del proceso de compilación.

FUENTE:

http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro32/31_funcion_del_analizador_sintctico.html

- **PRUEBAS DE QUE EL PROGRAMA FUNCIONA**



```
#Función para el primer ejercicio
def primerEjercicio(texto):
    pila = Pila()
    pila.push(NoTerminal(ElementoPila.SIMBOLO, "$0"))
    pila.mostrarPila()

    estado = ElementoPila.INICIAL
    d=2
    lexema = ""

    i = 0
    while(i<len(texto)):
        c = texto[i]

        if(estado == ElementoPila.INICIAL):
            if (es_Letra(c) or c == '_'):
                estado = ElementoPila.IDENTIFICADOR
                lexema += c
            elif (c == '+'):
                lexema += c
                pila.push(Terminal(ElementoPila.SIMBOLO, lexema+str(d)))
                d += 1
                estado = ElementoPila.INICIAL
                lexema = ""
                pila.mostrarPila()
            elif (c == '$'):
                pila.clear()
                nuevaPila = Pila()
                nuevaPila.push(Estado(ElementoPila.E, "$0E1"))
                nuevaPila.mostrarPila()
            else:
                print("ERROR")
```

```python
#Función para el segundo ejercicio
def segundoEjercicio(texto):

    pila = Pila()
    pila.push(NoTerminal(ElementoPila.SIMBOLO, "$0"))
    pila.mostrarPila()

    estado = ElementoPila.INICIAL
    d2 = 2
    d3 = 3
    lexema = ""

    #Inicia el automata del analizador
    i = 0
    while(i<len(texto)):
        c = texto[i]

        if(estado == ElementoPila.INICIAL):
            if (es_letra(c) or c == '_'): #Verifica si es letra o empieza con un "_"
                estado = ElementoPila.IDENTIFICADOR
                lexema += c
            elif (c == '+'):
                lexema += c
                pila.push(Terminal(ElementoPila.SIMBOLO, lexema + str(d3)))
                estado = ElementoPila.INICIAL
                lexema = ""
                pila.mostrarPila()
            elif (c == '$'):
                pila.clear()
                nuevaPila = Pila()
                nuevaPila.push(Estado(ElementoPila.E,"$0E1"))
                nuevaPila.mostrarPila()
```

Saldrá en pantalla el problema resuelto

```
El primer ejercicio en clase fue:
----------------------------
$0
$0 hola2
$0 hola2 +3
$0 hola2 +3 mundo4
$0E1
----------------------------
El segundo ejercicio en clase fue:
----------------------------
$0
$0 a2
$0 a2 +3
$0 a2 +3 b2
$0 a2 +3 b2 +3
$0 a2 +3 b2 +3 c2
$0 a2 +3 b2 +3 c2 +3
$0 a2 +3 b2 +3 c2 +3 d2
$0 a2 +3 b2 +3 c2 +3 d2 +3
$0 a2 +3 b2 +3 c2 +3 d2 +3 e2
$0 a2 +3 b2 +3 c2 +3 d2 +3 e2 +3
$0 a2 +3 b2 +3 c2 +3 d2 +3 e2 +3 f2
$0E1
----------------------------
PS C:\Users\Omar\OneDrive\Documentos\Escritorio\Septimo Semestre\SEM TRADUCTORES II\ANALIZADOR SINTACTICO (OBJ)>
```

- **CONCLUSIÓN**

La función principal del analizador sintáctico es garantizar que el código fuente cumpla con las reglas sintácticas del lenguaje de programación y preparar una representación interna adecuada para las etapas posteriores del proceso de

compilación o interpretación. Tuve un poco de problemas al realizar este programa ya que quise hacer interfaz, pero no he manejado ninguna, comencé a estudiar más sobre esto y me encantaría que en futuros programas emplearlos.