Analizador Léxico para Python Implementación con Flex/Lex

Arregoces, Gonzalez, Sanchez, Sharick
Laboratorio de Compiladores
Universidad del Norte
Barranquilla, Colombia

***Resumen*—Este documento presenta la implementación de un analizador léxico completo para el lenguaje de programación Python utilizando las herramientas Flex/Lex. El analizador reconoce más de 19 palabras reservadas, múltiples tipos de datos numéricos, operadores aritméticos, lógicos y de comparación, así como cadenas, comentarios y delimitadores. El sistema genera automáticamente archivos de salida que contienen los tokens reconocidos, una tabla de identificadores numerados y un contador de errores léxicos detectados durante el análisis.**

***Index Terms*—Compiladores, Análisis Léxico, Flex, Lex, Python, Tokens, Docker**

I. Introducción

El análisis léxico constituye la primera fase en el proceso de compilación de un programa, siendo responsable de la conversión del código fuente en una secuencia de tokens que serán posteriormente procesados por el analizador sintáctico. Este proyecto implementa un analizador léxico completo para Python utilizando Flex/Lex, herramientas ampliamente utilizadas en el desarrollo de compiladores.

El analizador desarrollado es capaz de reconocer la sintaxis léxica de Python, incluyendo palabras reservadas, identificadores, literales numéricos de diversos tipos, operadores, delimitadores y cadenas de texto. Adicionalmente, el sistema implementa mecanismos de detección de errores léxicos y genera reportes detallados del análisis realizado.

II. Descripción del Sistema

II-A. Características Principales

El analizador léxico implementado presenta las siguientes características:

- Reconocimiento de más de 19 palabras reservadas del lenguaje Python
- Soporte completo para tipos numéricos incluyendo enteros, reales, largos, imaginarios y notación científica
- Identificación de operadores aritméticos, de comparación, lógicos, bit a bit y de asignación
- Procesamiento de cadenas de texto, comentarios y delimitadores
- Detección y reporte de errores léxicos
- Generación automática de archivos de salida con tokens y tabla de identificadores

II-B. Palabras Reservadas

El analizador reconoce las siguientes categorías de palabras reservadas:

Control de flujo: `def`, `if`, `else`, `elif`, `for`, `while`, `break`, `continue`, `pass`, `return`

Operadores lógicos: `and`, `or`, `not`, `in`, `is`

Funciones: `import`, `print`, `range`, `len`

Valores booleanos: `True`, `False`

II-C. Tipos de Datos Numéricos

El sistema reconoce diversos formatos numéricos:

Enteros: Números enteros con o sin signo (ejemplo: `123`, `-456`, `+789`)

Largos: Enteros largos identificados con sufijo L o l (ejemplo: `40000L`, `-123l`)

Reales: Números de punto flotante y notación científica (ejemplo: `3.14`, `-2.5e-3`, `1.23E+4`)

Imaginarios: Números complejos con sufijo j (ejemplo: `5j`, `-3.2j`, `1+2j`)

II-D. Operadores

El analizador identifica los siguientes tipos de operadores:

Aritméticos: `+`, `-`, `*`, `/`, `%`, `**`, `//`

Comparación: `==`, `!=`, `<`, `>`, `<=`, `>=`, `<>`

Asignación: `=`, `+=`, `-=`, `*=`, `/=`, `//=`

Bit a bit: `&`, `|`, `^`, `~`, `«`, `»`

III. Requisitos del Sistema

III-A. Requisitos de Software

El sistema requiere las siguientes herramientas:

- Docker Desktop para Windows o Mac
- PowerShell o Command Prompt
- Flex/Lex (incluido en el contenedor Docker)
- Compilador C (incluido en el contenedor Docker)

IV. Instalación y Uso

IV-A. Instalación

El proceso de instalación consta de los siguientes pasos:

1. Clonar el repositorio:

```
git clone https://github.com/
  flaviofuego/
  Lab_Analisis_Lexico-Compiladores.git
cd Lab_Analisis_Lexico-Compiladores
```

2. Construir la imagen Docker:

```
docker build -t analizador-lexico .
```

3. Compilar el analizador:

```
docker run --rm -v "${PWD}:/workspace"
  analizador-lexico bash -c
  "make␣install-basic"
```

IV-B. Ejecución

Para ejecutar el analizador con el archivo de ejemplo:

```
docker run --rm -v "${PWD}:/workspace"
  analizador-lexico bash -c
  "make␣run-basic"
```

Para analizar un archivo específico:

```
docker run --rm -v "${PWD}:/workspace"
  analizador-lexico bash -c
  "./dist/LAB01_Arregoces_Gonzalez_
␣␣Sanchez_Oviedo␣./entradas/
␣␣archivo.py"
```

IV-C. Comandos Disponibles

El Makefile proporciona los siguientes comandos:

- `make install-basic`: Compila el analizador léxico
- `make run-basic`: Ejecuta análisis con archivo de ejemplo
- `make run-basic-file FILE=archivo.py`: Ejecuta análisis con archivo específico
- `make clean`: Limpia archivos generados
- `make help`: Muestra ayuda del sistema

V. Conclusiones

Se ha implementado exitosamente un analizador léxico completo para el lenguaje Python utilizando las herramientas Flex/Lex. El sistema es capaz de reconocer todos los elementos léxicos fundamentales del lenguaje, incluyendo palabras reservadas, identificadores, literales numéricos de diversos tipos, operadores y delimitadores.

La utilización de Docker como entorno de desarrollo garantiza la portabilidad y reproducibilidad del sistema en diferentes plataformas. La generación automática de archivos de salida facilita el análisis de los resultados y la detección de errores léxicos.

Este proyecto constituye una base sólida para la implementación de las fases subsecuentes del proceso de compilación, incluyendo el análisis sintáctico y semántico.