**Tecnológico de Costa Rica**

*Escuela de Ingeniería en Computación*

Texto

Descripción generada automáticamente

# Proyecto I “Análisis Léxico y Sintáctico”



*Compiladores e intérpretes*

Profesor:

Rodríguez Dávila, Allan

Presentado por:

Bonilla Espinoza, Alina [2016248502]

Lara Oses, Froylan [2018276191]

**Setiembre, 2023**

# Manual de usuario

Descarga de

instrucciones de compilación, ejecución y uso bien detalladas.

# Pruebas de funcionalidad

incluir screenshots.

# Descripción del problema

Se desea crear un nuevo lenguaje imperativo, ligero, que le permita realizar operaciones básicas para la configuración de chips, ya que esta es una industria que sigue creciendo constantemente, y cada vez estos chips necesitan ser configurados por lenguajes más ligeros y potentes. Es por esto por lo que este grupo de desarrolladores requiere desarrollar su propio lenguaje para el desarrollo de sistemas empotrados.

Este proyecto comprende la fase de análisis léxico y análisis sintáctico para la gramática descrita en la Tarea I. Se desarrollo un scanner utilizando la herramienta JFlex y un parser utilizando la herramienta Cup.

# Diseño del programa

A continuación, se detalla el proceso que se realizó para la creación del proyecto:

decisiones de diseño, algoritmos usados.

1. *Generación del Analizador Léxico:*

El archivo lexemas.cup se utiliza como entrada para Java Cup, que genera un analizador léxico Java basado en las reglas y patrones definidos en lexemas.cup. El analizador léxico generado tiene la capacidad de escanear y reconocer tokens en el código fuente.

1. *Análisis del Código Fuente:*

El programa AnalizadorLexico.java se encarga de abrir un archivo de código fuente o de recibir texto de entrada y lo pasa al analizador léxico generado. El analizador léxico escanea el código fuente identificando y clasificando cada token según las reglas definidas en lexemas.cup.

1. *Etiquetado de Tokens:*

Para cada token reconocido, el analizador léxico asigna un tipo de token de acuerdo con la enumeración Tokens.java. Esto permite que se etiquete cada parte del código fuente con un tipo específico (identificador, palabra clave, operador, etc.).

1. *Generación de Resultados:*

Se generar una lista con todos los tokens encontrados y sus tipos.

1. *Manejo de Errores:*

Si el analizador léxico encuentra caracteres o secuencias que no coinciden con ninguno de los patrones definidos en lexemas.cup, se generará un token de tipo ERROR para señalar el error léxico.

# Librerías usadas

Para la elaboración de este proyecto se hizo uso de las siguientes librería:

* File: Se utiliza para representar rutas de archivos.
* FileReader: Se utiliza para leer archivos.
* FileWriter: Se utiliza para la escritura de archivos.

# Análisis de resultados

## **Objetivos alcanzados**

* Todos los objetivos solicitados en la tarea fueron desarrollados (del inciso a hasta el q), se realizaron con éxito.

**Objetivos no logrados**

* Todos los objetivos solicitados fueron alcanzados

# Bitácora

Seguidamente, se muestra la bitácora recaudada del archivo compartido en la plataforma de GitHub de los integrantes: Froylan Lara (Udue11) y Alina Bonilla (Alina-bonilla).

# Puntos extra