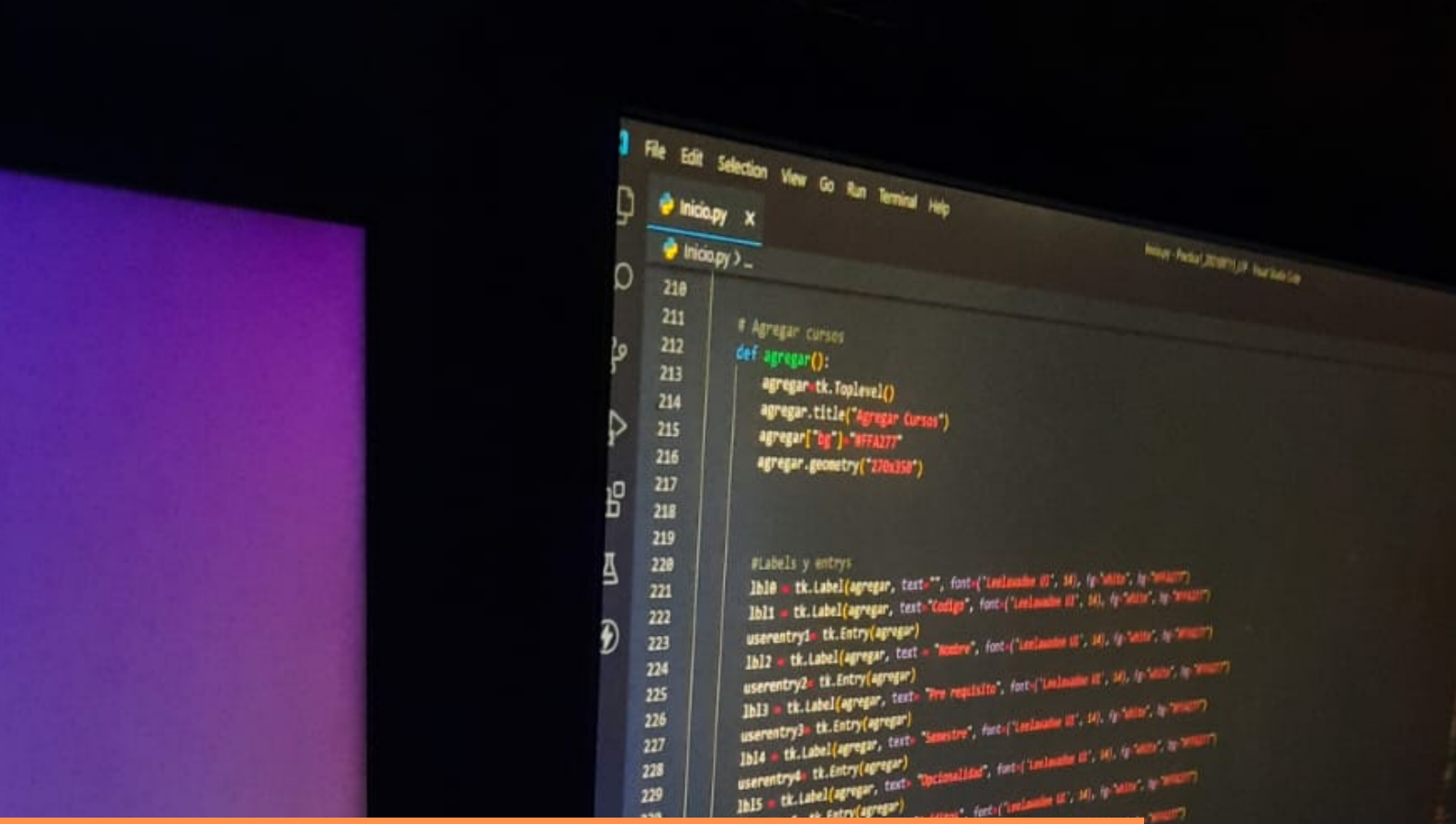


MANUAL TÉCNICO

Universidad San Carlos de Guatemala
Organización de Lenguajes y Compiladores 1
Pablo Josué Barahona Luncey
202109715
3560855890101

PROYECTO 1

Acerca del programa

Este programa fue desarrollado con el lenguaje de programación JAVA como base principal, utilizando su interfaz. También se usó JFLEX para crear el analizador léxico y CUP para el analizador sintáctico,

Este programa cuenta con diferentes clases (explicación en las siguientes páginas) y el usó de librerías nativas de JAVA.

El objetivo principal del programa es crear un analizador léxico y sintáctico que sea capaz de reconocer una gramática. Posterior a eso, utilizar los datos obtenidos y crear reportes gráficos con Graphviz y archivos JSON.

Clases y Métodos

Clase Interface

Esta es la clase principal del archivo, es quien tiene la interfaz gráfica, las funcionalidades de los botones y llama a los métodos de las demás clases.

Archivo lexico.jflex

En este archivo se tienen establecidas las gramáticas a usar en el analizador. Su objetivo es tener las expresiones regulares que se usan en el CUP.

Archivo sintáctico.cup

En este archivo se tiene la estructura a utilizar. Con este archivo generamos el analizador sintáctico.

Clase Generador

Esta clase utiliza los archivos JFLEX y CUP y los genera en un archivo JAVA, para que posteriormente los podamos usar en nuestra clase principal.

Clase arbol

Esta es nuestra clase nodo, recibe los datos y los devuelve al archivo AFD.

Clases y Métodos

Clase AFD

Esta es la clase más importante, ya que acá se hace la mayoría de los métodos para analizar los archivos y generar los reportes, los métodos que se tienen son los siguientes:

- asignar(): En este método asignamos los valores que se leen a lo que necesitamos
- metodo(): Este método genera las funciones del árbol, los First and Post, el número de hoja y la anulabilidad del nodo.
- crear(): A través de este método, creamos el archivo graphviz y este mismo genera el archivo dot y la imagen JPG

Clase Excepcion

Esta clase actúa como un nodo, ya que nos devuelve los errores que generamos en la lectura del archivo

Interface

Acá al ser nuestra clase principal, tenemos el manejo de los botones y las demás funcionalidades, entre las funcionalidades principales está:

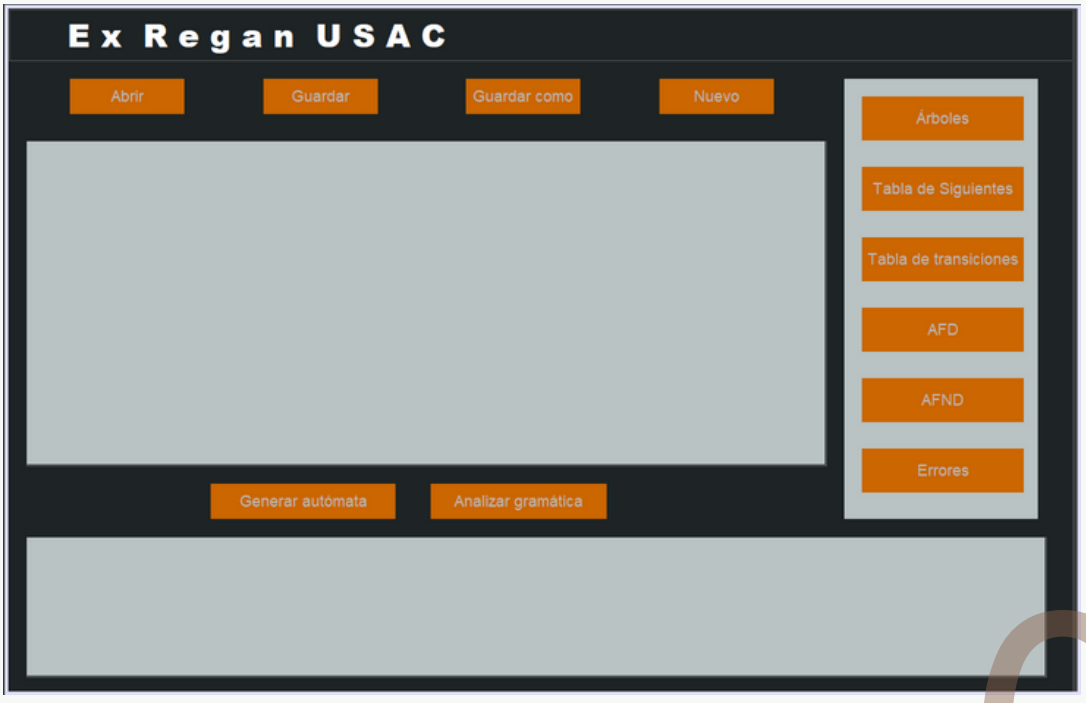
btngenerar()

En este función llamamos al archivo lexer y sintaxis, para poder enviar los datos y posteriormente recibirlos y proyectarlos. Este método también analiza los errores y genera el reporte.

btnanalizar()

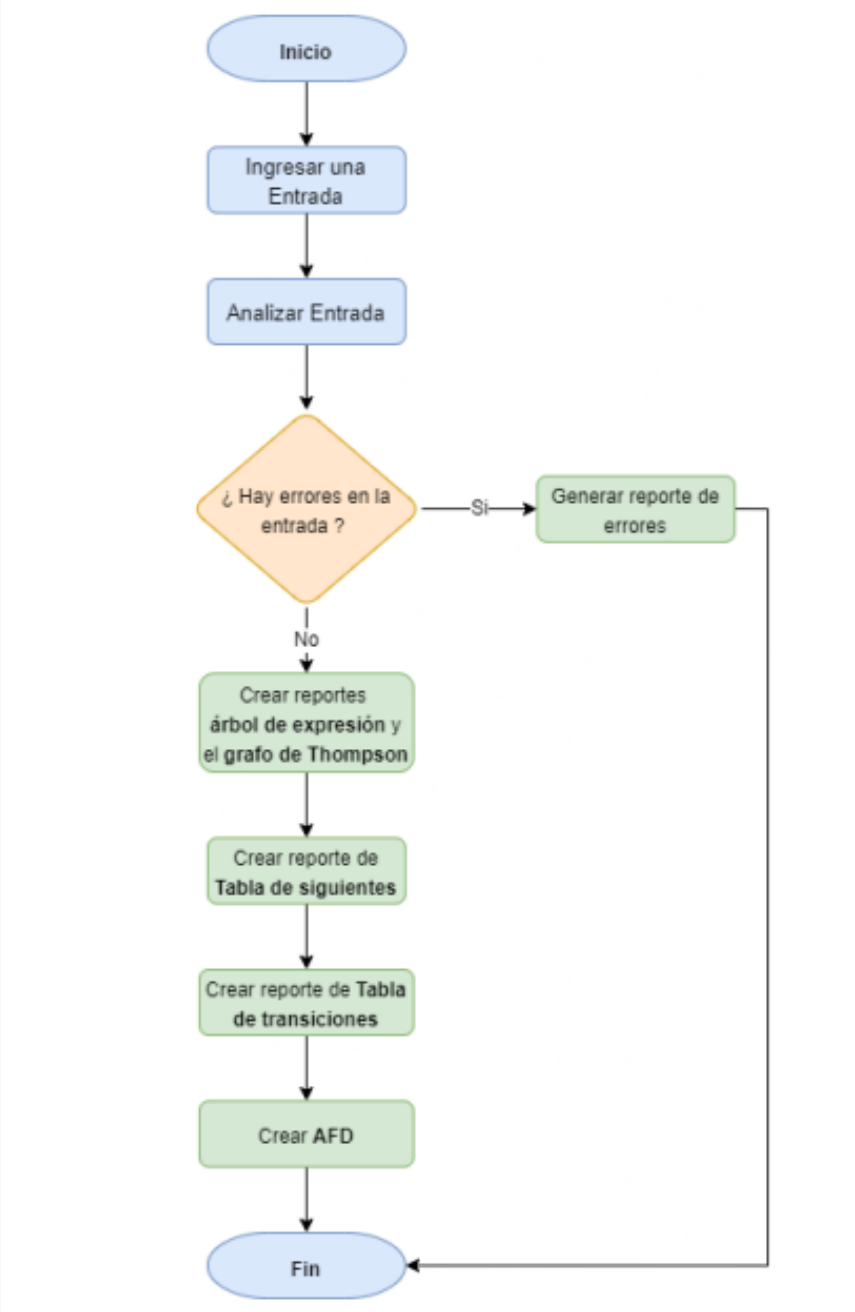
En esta función se genera el archivo JSON y se evalúa la cadena propuesta

Finalmente, la interfaz se muestra así:



Diagramas de flujo de apoyo

Generación de AFD



Diagramas de flujo de apoyo

Validación de cadena