

Manual de programador para el analizador léxico

Materia: Lenguaje de autómatas I

Ingeniería en sistemas computacionales

Pérez Castro Julio de Jesús

Sosa Hernández Luis Enrique

Profesor: Hernández Silva José

Noviembre del 2016

Introducción

En este manual de programador redactaremos las clases de la interfaz del usuario, daremos a conocer los pasos para generar el analizador léxico de la manera más clara posible para que el lector (programador) cuando quiera editar o saber más a fondo la funcionalidad del código este documento le sea de ayuda.

Explicaremos los comandos necesarios para generar los archivos en lenguaje C para la creación del analizador léxico en Byson, como editarlos para que imprima tokens o para que el lenguaje C retorne los token al momento de ser requeridos en la interfaz gráfica de JAVA.

Clases

Abrir

Esta clase fue desarrollada mediante las herramientas de diseño de JFrame para la creación de botones, etiquetas, etc.

Mediante el uso de las siguientes bibliotecas:

* JFileChooser: Nos presenta una ventana de herramientas, que dependiendo de nuestra lógica, nos permitirá Abrir o Guardar archivos
* JOptionPane: Nos muestra mensajes o nos permite ingresar campos de texto.
* filechooser.FileFilter: Permite la filtración de archivos.
* filechooser.FileNameExtensionFilter: Filtra extensiones de archivos.

Gracias a los métodos de estas clases y a los métodos de botón es posible abrir archivos, según nuestros requerimientos.

GUICompilador

Es la clase principal del main, que nos permite hacer uso de las demás clases y hace una instanciación del frame principal.

Nuevo

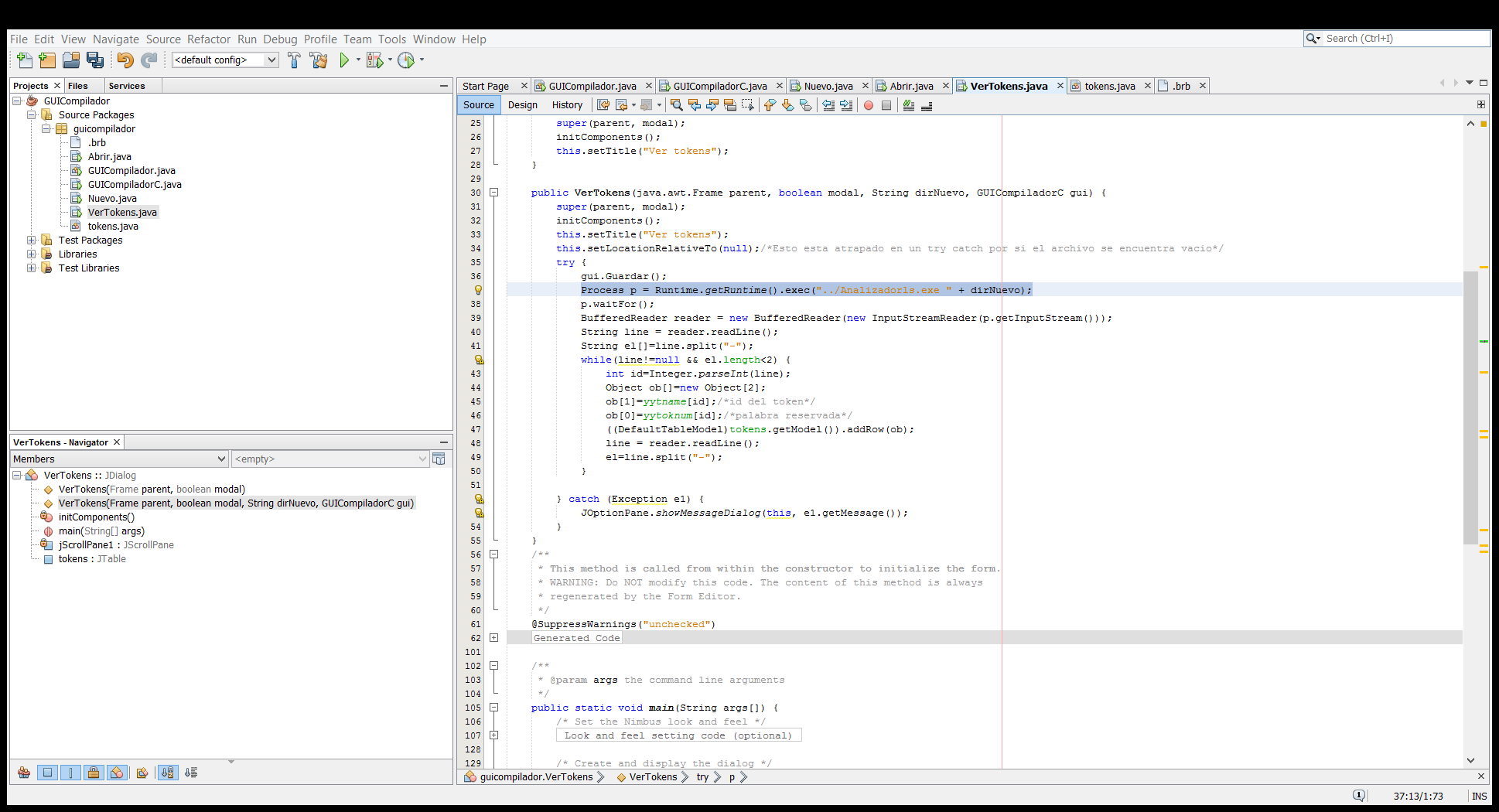
Esta clase nos permite crear un nuevo archivo con ayuda de los métodos de las siguientes bibliotecas:

* java.io.DataInputStream;
* java.io.FileInputStream;
* javax.swing.JFileChooser;
* javax.swing.JOptionPane;

Los métodos de estas clases nos permiten escribir archivos de texto, así como elegir donde guardarlo, ingresar su nombre.

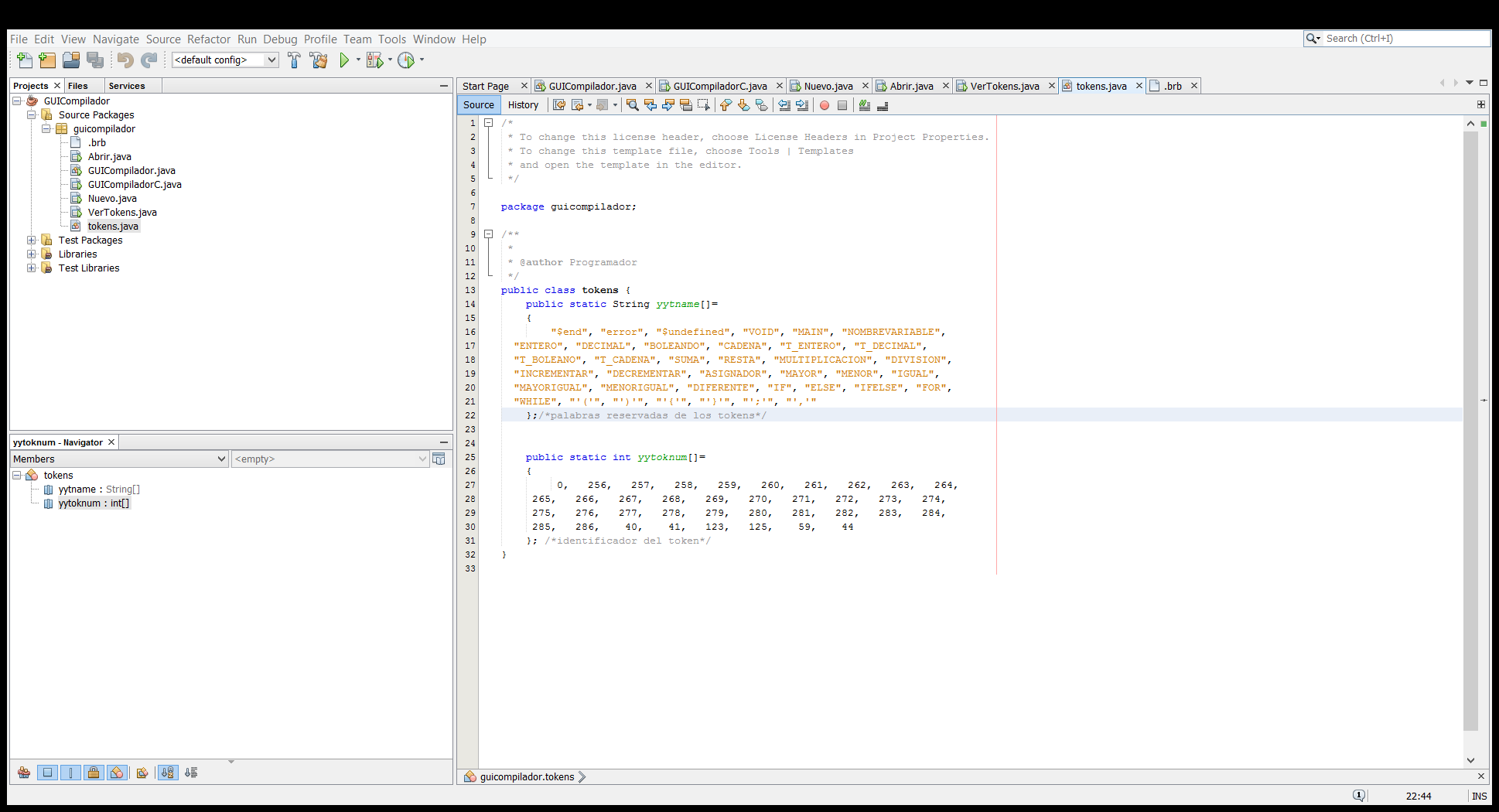
VerTokens

Esta clase nos permite visualizar los tokens, leyendo lo que retorna el analizador escrito en C, gracias a los métodos de las siguientes clases – bibliotecas, todas son guardadas en una tabla para su posterior visualización. Esta clase hace uso de la clase tokens.

* import static guicompilador.tokens.\*;
* java.io.BufferedReader;
* java.io.InputStreamReader;
* javax.swing.JOptionPane;
* javax.swing.table.DefaultTableModel;

Con esta linea de codigo se recuperan la impresión del Id del token en el ejecutable de C.

Tokens

En esta clase se crean dos arrays que son los generados por el programa Byson que representan a los tokens.

Esta clase fue creada debido a que el analizador escrito en C, lo unico que retorna es el Id del token requerido y para mostrar el token es necesario enlazarla con otra matriz donde su posicion coincida con el Id de la otra matriz.

GUIcompilador.c

Esta es la clase de la interfaz principal donde están implementados todos los componentes que hacen uso de los métodos de las otras clases. Esta interfaz representa la comodidad de uso para el usuario para la creación, para la lectura y para guardar archivos y para la visualización de los tokens, gracias a los eventos de los botones.

Creación del analizador léxico

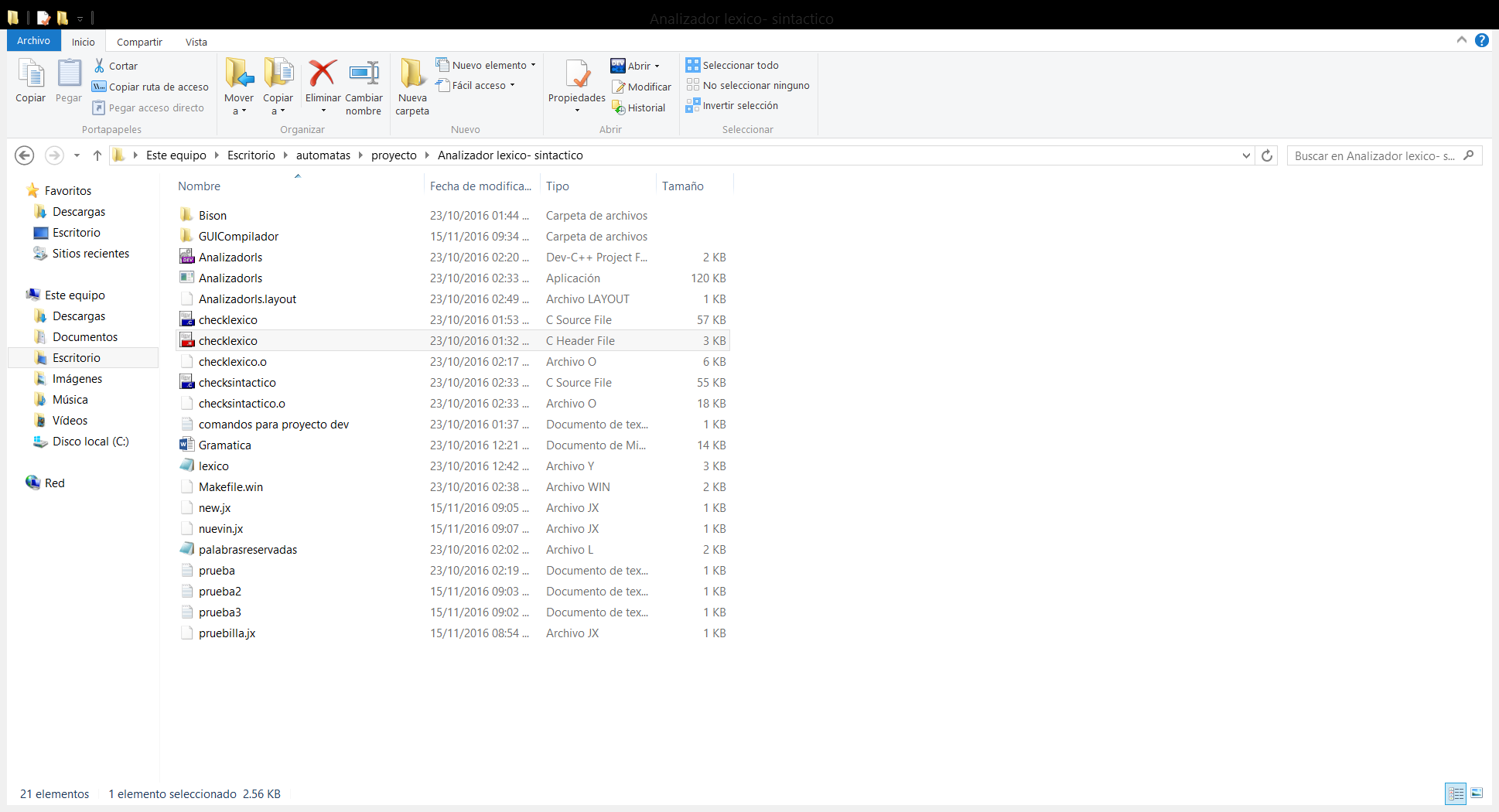
En el archivo léxico.y está implícita la gramática libre de contexto y los tokens para el lenguaje c-copia.

Haciendo uso del programa Byson podemos generar el analizador léxico escrito en lenguaje C, para llevar acabo esto es necesario de la línea de consola (CMD) con el siguiente comando:

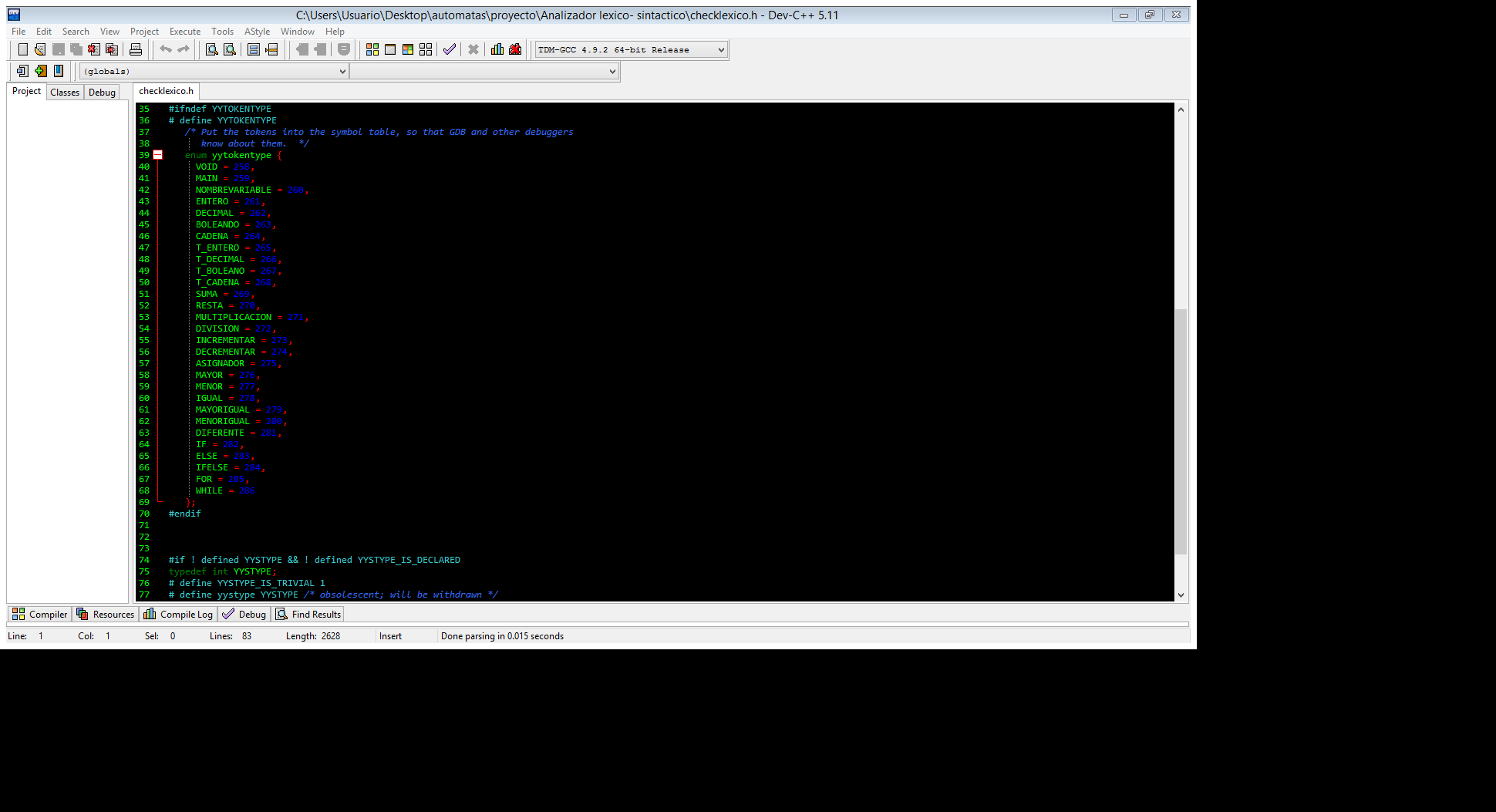
bison -d -oanalexico.c lexico.y

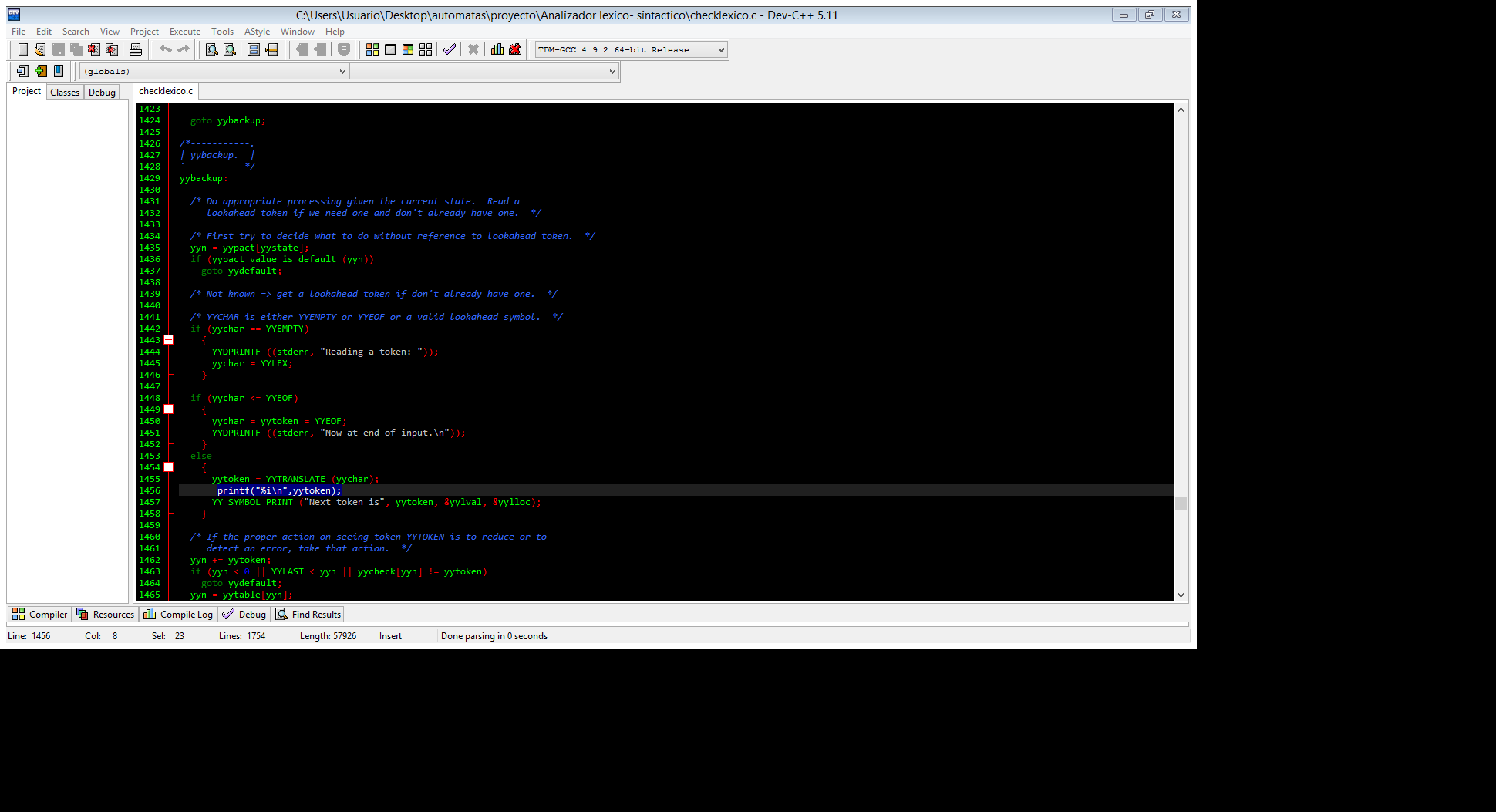
El archivo léxico.y debe estar en la misma carpeta donde se encuentre el programa Byson.

Despues de realizar esta acción se crean dos archivos de C, un archivo .h y un archivo .c.



Donde en el archivo .h se encuentran todos los tokens y el archivo .c se encuentra la sintaxis necesaria para reconocer dichos tokens.



En el archivo .c es necesario editar para que muestre en consola el Id de los tokens y poder recuperarlos en la interfaz grafica de JAVA.

La interfaz grafica fue tomada del usuario de Youtube:

AllTechnology – JC

Que humildemente dona codigo para la comunidad OpenSource.