GRUPO E04

TEORÍA DE AUTÓMATAS Y COMPUTACIÓN

PRÁCTICA

ANALIZADOR LÉXICO JFLEX & CUP

**ÍNDICE**

[**1. Información de elementos léxicos:** 2](#_Toc70934889)

[**2. Manual de usuario:** 4](#_Toc70934890)

[**3. Código Jflex** 5](#_Toc70934891)

[**4. Código Cup** 8](#_Toc70934892)

[**5. Explicación:** 9](#_Toc70934893)

# **1. Información de elementos léxicos:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Token** | **Descripción** | **Expresión regular** | **Ejemplo** |
| BUCLE | Reconoce las directivas de cada  bucle | while | for | do | while |
| NUMERO | Reconoce números enteros sin signos | [0-9]+ | 45 |
| VARIABLE | Reconoce cadenas de caracteres como nombres de variables | [A-Za-z][a-zA-Z0-9\_]\* | Ac\_D9 |
| TRUE | Reconoce la directiva true | true | true |
| FALSE | Reconoce la directiva false | false | false |
| ENTERO | Reconoce la directiva int | int | int |
| BOOLEAN | Reconoce la directiva boolean | boolean | boolean |
| NORETORNO | Reconoce la directiva void | void | void |
| RETORNO | Reconoce la directiva return | return | return |
| SUMA | Reconoce el símbolo suma | + | + |
| RESTA | Reconoce el símbolo resta | - | - |
| POR | Reconoce el símbolo por | \* | \* |
| ENTRE | Reconoce el símbolo  entre | / | / |
| RELACIONAL | Reconoce los símbolos  relacionales de las expresiones | <|<=|>|>=|==|!= | < |
| LOGICA | Reconoce los símbolos lógicos de las expresiones | &&| || |! | && |
| PARENTESIS | Reconoce el paréntesis de  apertura y cierre en una expresión | ( | ) | ( |
| LLAVE | Reconoce la llave abierta y cerrada al  comienzo y al final  de un método o  clase | { | } | { |
| PUNTUACIÓN | Reconoce símbolos de puntuación | , | ; | . | ; |
| CLASE | Reconoce la directiva class | class | class |
| PUBLIC | Reconoce la directiva public | public | public |
| STATIC | Reconoce la directiva static | static | static |
| INCREMENTO | Reconoce una variable que se incrementa | Variable ++ | variable += | Var++ |
| DECREMENTO | Reconoce una variable que se decrementa | Variable -- | variable  -= | Var-- |
| ASIGNACION | Reconoce el símbolo de asignación | = | = |
| SALTODELINEA | Reconoce los saltos de línea | \n|\r|\r\n |  |
| ESPACIO | Reconoce los espacios en blanco | “ ” |  |
| MAIN | Reconoce la cabecera de los métodos main | main( | main( |

# **2. Manual de usuario:**

# **3. Código Jflex**

/\* Sección de declaraciones de JFlex \*/

**import java\_cup.runtime.Symbol;**

**%%**

**%class AnalizadorLexicoCup**

**%type java\_cup.runtime.Symbol**

**%cup**

**%full**

**%line**

**%char**

/\* Inicio de Expresiones regulares \*/

**NUMERO = [0-9]+**

**ENTERO = "int"**

**BOOLEAN = "boolean"**

**RETORNO = "return"**

**NORETORNO = "void"**

**VARIABLE = [a-zA-Z] [a-zA-Z0-9\_]\***

**SALTODELINEA = \n|\r|\r\n**

**ASIGNACION = "="**

**PUNTUACION = ";" | "," |"."**

**BUCLE = "for" | "while" | "do"**

**SUMA = "+"**

**RESTA = "-"**

**POR = "\*"**

**ENTRE = "/"**

**RELACIONAL = "<"|"<="|">"|">="|"=="|"!="**

**LOGICA = "&&"|"||"|"!"**

**ESPACIO = " "**

**PARENTESIS = "(" | ")"**

**LLAVE = "{" | "}"**

**VERDADERO = "True"**

**FALSO = "False"**

**MAIN = "main("**

**PUBLIC = "public"**

**STATIC = "static"**

**CLASS = "class"**

/\* Finaliza expresiones regulares \*/

%%

/\* Finaliza la sección de declaraciones de JFlex \*/

/\* Inicia sección de reglas \*/

// Cada regla está formada por una {expresión} espacio {código}

**{NUMERO} { return new Symbol(sym.NUMERO, yytext()); }**

**{SALTO} { }**

**{ASIGNACION} { return new Symbol(sym.ASIGNACION, yytext()); }**

**{PUNTUACION} { return new Symbol(sym.PUNTUACION, yytext()); }**

**{BUCLE} { return new Symbol(sym.BUCLE, yytext()); }**

**{SUMA} { return new Symbol(sym.SUMA, yytext()); }**

**{RESTA} { return new Symbol(sym.RESTA, yytext()); }**

**{POR} { return new Symbol(sym.POR, yytext()); }**

**{ENTRE} { return new Symbol(sym.ENTRE, yytext()); }**

**{LOGICA} { return new Symbol(sym.LOGICA, yytext()); }**

**{RELACIONAL} { return new Symbol(sym.RELACIONAL, yytext());}**

**{ESPACIO} { }**

**{ENTERO} { return new Symbol(sym.ENTERO, yytext()); }**

**{BOOLEAN} { return new Symbol(sym.BOOLEAN, yytext()); }**

**{RETORNO} { return new Symbol(sym.RETORNO, yytext()); }**

**{NORETORNO} { return new Symbol(sym.NORETORNO, yytext()); }**

**{CLASS} { return new Symbol(sym.CLASS, yytext()); }**

**{STATIC} { return new Symbol(sym.STATIC, yytext()); }**

**{PUBLIC} { return new Symbol(sym.PUBLIC, yytext()); }**

**{PARENTESIS} { return new Symbol(sym.PARENTESIS, yytext()); }**

**{LLAVE} { return new Symbol(sym.LLAVE, yytext()); }**

**{VERDADERO} { return new Symbol(sym.VERDADERO, yytext()); }**

**{FALSO} { return new Symbol(sym.FALSO, yytext()); }**

**{MAIN} { return new Symbol(sym.MAIN, yytext()); }**

**{VARIABLE} { return new Symbol(sym.VARIABLE, yytext()); }**

/\* Finaliza la sección de reglas\*/

# **4.Código Cup**

import java\_cup.runtime.\*;

import java.io;

parser code {:

    public static void main (String args[]) throws Exception{

        FileInputStream f = *new* FileInputStream(args[0]);

        DataInputStream r = *new* DataInputStream(f);

*try*{

*new* parser(*new* Yylex(r)).parse();

        }

*catch* (Exception e){

            System.out.println("Análisis incorrecto");

            System.exit(1);

        }

        System.out.println("Análisis correcto");

    }

*:*}

terminal NUMERO, ENTERO, BOOLEAN, RETORNO, NORETORNO, VARIABLE, SALTODELINEA, ASIGNACION, PUNTUACION, BUCLE, SUMA, RESTA, POR, ENTRE, RELACIONAL, LOGICA, ESPACIO, PARENTESIS, LLAVE, VERDADERO, FALSO, MAIN, PUBLIC, STATIC, CLASS;

non terminal incremento, decremento, metodo, clase, principal, llamada;

precedence left SUMA, RESTA;

precedence left POR, ENTRE;

incremento *::*= VARIABLE SUMA SUMA;

decremento *::*= VARIABLE RESTA RESTA;

metodo *::*= PUBLIC STATIC ENTERO VARIABLE | PUBLIC STATIC BOOLEAN VARIABLE | PUBLIC STATIC NORETORNO VARIABLE;

clase *::*= PUBLIC CLASS;

principal *::*= PUBLIC STATIC CLASS MAIN;

llamada *::*= VARIABLE PARENTESIS;

# **5.Explicación:**

El programa reconoce una serie de expresiones léxicas que son palabras o secuencias de palabras (en los casos de las clases y declaraciones de métodos). Las explicaciones de cada expresión léxica se encuentran en la tabla del comienzo.