// Código para incializar

%init{

tokenpos=0;

cant\_coment=0;

%init}

// Definicion

letra [a-z,A-Z]

%%

//declaración de las ETL’s

(\ |\t)+ {tokenpos+=yylength();break;}

\n {tokenpos=0;break;}

"\\*" {cant\_coment++;tokenpos+=yylength();break;}

"\*/" {cant\_coment--;tokenpos+=yylength();break;}

"float"{tokenpos+=yylength();System.out.println("float");break;}

"int" {tokenpos+=yylength();System.out.println("int");break;}

"break" {tokenpos+=yylength();System.out.println("break");break;}

"continue"

{tokenpos+=yylength();System.out.println("continue");break;}

"else" {tokenpos+=yylength();System.out.println("else");break;}

"if" {tokenpos+=yylength();System.out.println("if");break;}

"return"

{tokenpos+=yylength();System.out.println("return");break;}

"while" {tokenpos+=yylength();System.out.println("while");break;}

"+" {tokenpos+=yylength();System.out.println("+");break;}

"-" {tokenpos+=yylength();System.out.println("-");break;}

"\*" {tokenpos+=yylength();System.out.println("\*");break;}

"/" {tokenpos+=yylength();System.out.println("/");break;}

"%" {tokenpos+=yylength();System.out.println("%");break;}

"!" {tokenpos+=yylength();System.out.println("!");break;}

"?" {tokenpos+=yylength();System.out.println("?");break;}

":" {tokenpos+=yylength();System.out.println(":");break;}

"=" {tokenpos+=yylength();System.out.println("=");break;}

"," {tokenpos+=yylength();System.out.println(",");break;}

">" {tokenpos+=yylength();System.out.println(">");break;}

"<" {tokenpos+=yylength();System.out.println("<");break;}

"(" {tokenpos+=yylength();System.out.println("(");break;}

")" {tokenpos+=yylength();System.out.println(")");break;}

"{" {tokenpos+=yylength();System.out.println("llaveO");break;}

"}" {tokenpos+=yylength();System.out.println("llaveC");break;}

"||" {tokenpos+=yylength();System.out.println("OR");break;}

"&&" {tokenpos+=yylength();System.out.println("AND");break;}

"==" {tokenpos+=yylength();System.out.println("==");break;}

";" {tokenpos+=yylength();System.out.println("puntoYc");break;}

{letra} ({letra}|{digito})\* {System.out.println("Identificador: "+yytext());break;}

INIT{intval=0;} ([0-9]

ACTION{tokenpos++; Integer aux=new

Integer(yytextchar()+"");

intval=intval\*10 + aux.intValue();} ) +

{System.out.println("Natural: "+intval);break;}

}

INIT{realval1=0;realval2=0;cantreal=1;}

([0-9] ACTION{tokenpos++; Integer aux=new

Integer(yytextchar()+"");

realval1=realval1\*10 + aux.intValue();} ) +\. ([0-9]

ACTION{tokenpos++; Integer aux=new

Integer(yytextchar()+"");

cantreal=cantreal\*10;

realval2=realval2\*10 + aux.intValue();}) +

{real=realval1+(realval2/cantreal); System.out.println("Real:"+real);break;}

. {System.out.println("Error el Identificador no es valido" +yytext()) ;

break;}

%%

/\* sección código del usuario \*/

import java.io.\*;

import java.lang.System;

import static javafx.scene.input.KeyCode.L;

import jdk.nashorn.internal.parser.Lexer;

class Main

{

public static void main (String argv[])

{

Lexer P = null;

try { P = new Lexer(argv[0]);

}catch(Exception e){}

int N = 0;

while (N!=-1)

{

N = P.next\_token();

System.out.print("Expresion regular nro :"+N+" - Lexema :" + L.yytext());

}

System.out.println("Finalizacion: C-- ");

}

}